OpenShift Container Platform 4.6

Installing

Installing and configuring OpenShift Container Platform clusters
Installing and configuring OpenShift Container Platform clusters
Abstract

This document provides information about installing OpenShift Container Platform and details about some configuration processes.
2.3.10. Next steps

2.4. INSTALLING A CLUSTER ON AWS WITH CUSTOMIZATIONS

2.4.1. Prerequisites

2.4.2. Internet access for OpenShift Container Platform

2.4.3. Generating an SSH private key and adding it to the agent

2.4.4. Obtaining the installation program

2.4.5. Creating the installation configuration file

2.4.5.1. Installation configuration parameters

2.4.5.1.1. Required configuration parameters

2.4.5.1.2. Network configuration parameters

2.4.5.1.3. Optional configuration parameters

2.4.5.1.4. Optional AWS configuration parameters

2.4.5.2. Sample customized install-config.yaml file for AWS

2.4.5.3. Configuring the cluster-wide proxy during installation

2.4.6. Deploying the cluster

2.4.7. Installing the OpenShift CLI by downloading the binary

2.4.7.1. Installing the OpenShift CLI on Linux

2.4.7.2. Installing the OpenShift CLI on Windows

2.4.7.3. Installing the OpenShift CLI on macOS

2.4.8. Logging in to the cluster by using the CLI

2.4.9. Logging in to the cluster by using the web console

2.4.10. Telemetry access for OpenShift Container Platform

2.4.11. Next steps

2.5. INSTALLING A CLUSTER ON AWS WITH NETWORK CUSTOMIZATIONS

2.5.1. Prerequisites

2.5.2. Internet access for OpenShift Container Platform

2.5.3. Generating an SSH private key and adding it to the agent

2.5.4. Obtaining the installation program

2.5.5. Network configuration phases

2.5.6. Creating the installation configuration file

2.5.6.1. Installation configuration parameters

2.5.6.1.1. Required configuration parameters

2.5.6.1.2. Network configuration parameters

2.5.6.1.3. Optional configuration parameters

2.5.6.1.4. Optional AWS configuration parameters

2.5.6.2. Sample customized install-config.yaml file for AWS

2.5.6.3. Configuring the cluster-wide proxy during installation

2.5.7. Cluster Network Operator configuration

2.5.7.1. Cluster Network Operator configuration object

defaultNetwork object configuration

Configuration for the OpenShift SDN CNI cluster network provider

Configuration for the OVN–Kubernetes CNI cluster network provider

kubeProxyConfig object configuration

2.5.8. Specifying advanced network configuration

2.5.9. Configuring an Ingress Controller Network Load Balancer on a new AWS cluster

2.5.10. Configuring hybrid networking with OVN–Kubernetes

2.5.11. Deploying the cluster

2.5.12. Installing the OpenShift CLI by downloading the binary

2.5.12.1. Installing the OpenShift CLI on Linux

2.5.12.2. Installing the OpenShift CLI on Windows

2.5.12.3. Installing the OpenShift CLI on macOS

2.5.13. Logging in to the cluster by using the CLI

2.5.14. Logging in to the cluster by using the web console
2.5.15. Telemetry access for OpenShift Container Platform
2.5.16. Next steps

2.6. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK
2.6.1. Prerequisites
2.6.2. About installations in restricted networks
  2.6.2.1. Additional limits
2.6.3. About using a custom VPC
  2.6.3.1. Requirements for using your VPC
  2.6.3.2. VPC validation
  2.6.3.3. Division of permissions
  2.6.3.4. Isolation between clusters
2.6.4. Internet access for OpenShift Container Platform
2.6.5. Generating an SSH private key and adding it to the agent
2.6.6. Creating the installation configuration file
  2.6.6.1. Installation configuration parameters
    2.6.6.1.1. Required configuration parameters
    2.6.6.1.2. Network configuration parameters
    2.6.6.1.3. Optional configuration parameters
    2.6.6.1.4. Optional AWS configuration parameters
  2.6.6.2. Sample customized install-config.yaml file for AWS
  2.6.6.3. Configuring the cluster-wide proxy during installation
2.6.7. Deploying the cluster
2.6.8. Installing the OpenShift CLI by downloading the binary
  2.6.8.1. Installing the OpenShift CLI on Linux
  2.6.8.2. Installing the OpenShift CLI on Windows
  2.6.8.3. Installing the OpenShift CLI on macOS
2.6.9. Logging in to the cluster by using the CLI
2.6.10. Disabling the default OperatorHub sources
2.6.11. Telemetry access for OpenShift Container Platform
2.6.12. Next steps

2.7. INSTALLING A CLUSTER ON AWS INTO AN EXISTING VPC
2.7.1. Prerequisites
2.7.2. About using a custom VPC
  2.7.2.1. Requirements for using your VPC
  2.7.2.2. VPC validation
  2.7.2.3. Division of permissions
  2.7.2.4. Isolation between clusters
2.7.3. Internet access for OpenShift Container Platform
2.7.4. Generating an SSH private key and adding it to the agent
2.7.5. Obtaining the installation program
2.7.6. Creating the installation configuration file
  2.7.6.1. Installation configuration parameters
    2.7.6.1.1. Required configuration parameters
    2.7.6.1.2. Network configuration parameters
    2.7.6.1.3. Optional configuration parameters
    2.7.6.1.4. Optional AWS configuration parameters
  2.7.6.2. Sample customized install-config.yaml file for AWS
  2.7.6.3. Configuring the cluster-wide proxy during installation
2.7.7. Deploying the cluster
2.7.8. Installing the OpenShift CLI by downloading the binary
  2.7.8.1. Installing the OpenShift CLI on Linux
  2.7.8.2. Installing the OpenShift CLI on Windows
  2.7.8.3. Installing the OpenShift CLI on macOS
2.7.9. Logging in to the cluster by using the CLI
2.7.10. Logging in to the cluster by using the web console
2.7.11. Telemetry access for OpenShift Container Platform
2.7.12. Next steps

2.8. INSTALLING A PRIVATE CLUSTER ON AWS
2.8.1. Prerequisites
2.8.2. Private clusters
2.8.2.1. Private clusters in AWS
2.8.2.1.1. Limitations
2.8.3. About using a custom VPC
2.8.3.1. Requirements for using your VPC
2.8.3.2. VPC validation
2.8.3.3. Division of permissions
2.8.3.4. Isolation between clusters
2.8.4. Internet access for OpenShift Container Platform
2.8.5. Generating an SSH private key and adding it to the agent
2.8.6. Obtaining the installation program
2.8.7. Manually creating the installation configuration file
2.8.7.1. Installation configuration parameters
2.8.7.1.1. Required configuration parameters
2.8.7.1.2. Network configuration parameters
2.8.7.1.3. Optional configuration parameters
2.8.7.1.4. Optional AWS configuration parameters
2.8.7.2. Sample customized install-config.yaml file for AWS
2.8.7.3. Configuring the cluster-wide proxy during installation
2.8.8. Deploying the cluster
2.8.9. Installing the OpenShift CLI by downloading the binary
2.8.9.1. Installing the OpenShift CLI on Linux
2.8.9.2. Installing the OpenShift CLI on Windows
2.8.9.3. Installing the OpenShift CLI on macOS
2.8.10. Logging in to the cluster by using the CLI
2.8.11. Logging in to the cluster by using the web console
2.8.12. Telemetry access for OpenShift Container Platform
2.8.13. Next steps

2.9. INSTALLING A CLUSTER ON AWS INTO A GOVERNMENT REGION
2.9.1. Prerequisites
2.9.2. AWS government regions
2.9.3. Private clusters
2.9.3.1. Private clusters in AWS
2.9.3.1.1. Limitations
2.9.4. About using a custom VPC
2.9.4.1. Requirements for using your VPC
2.9.4.2. VPC validation
2.9.4.3. Division of permissions
2.9.4.4. Isolation between clusters
2.9.5. Internet access for OpenShift Container Platform
2.9.6. Generating an SSH private key and adding it to the agent
2.9.7. Obtaining the installation program
2.9.8. Manually creating the installation configuration file
2.9.8.1. Installation configuration parameters
2.9.8.1.1. Required configuration parameters
2.9.8.1.2. Network configuration parameters
2.9.8.1.3. Optional configuration parameters
2.9.8.1.4. Optional AWS configuration parameters
2.9.8.2. Sample customized install-config.yaml file for AWS
2.9.8.3. AWS regions without a published RHCOS AMI
2.9.8.4. Uploading a custom RHCOS AMI in AWS
2.9.8.5. Configuring the cluster-wide proxy during installation
2.9.9. Deploying the cluster
2.9.10. Installing the OpenShift CLI by downloading the binary
  2.9.10.1. Installing the OpenShift CLI on Linux
  2.9.10.2. Installing the OpenShift CLI on Windows
  2.9.10.3. Installing the OpenShift CLI on macOS
2.9.11. Logging in to the cluster by using the CLI
2.9.12. Logging in to the cluster by using the web console
2.9.13. Telemetry access for OpenShift Container Platform
2.9.14. Next steps
2.10. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN AWS BY USING CLOUDFORMATION TEMPLATES
  2.10.1. Prerequisites
  2.10.2. Internet access for OpenShift Container Platform
  2.10.3. Required AWS infrastructure components
    2.10.3.1. Cluster machines
    2.10.3.2. Other infrastructure components
    2.10.3.3. Certificate signing requests management
    2.10.3.4. Required AWS permissions
  2.10.4. Obtaining the installation program
  2.10.5. Generating an SSH private key and adding it to the agent
  2.10.6. Creating the installation files for AWS
    2.10.6.1. Optional: Creating a separate /var partition
    2.10.6.2. Creating the installation configuration file
    2.10.6.3. Configuring the cluster-wide proxy during installation
    2.10.6.4. Creating the Kubernetes manifest and Ignition config files
  2.10.7. Extracting the infrastructure name
  2.10.8. Creating a VPC in AWS
    2.10.8.1. CloudFormation template for the VPC
  2.10.9. Creating networking and load balancing components in AWS
    2.10.9.1. CloudFormation template for the network and load balancers
  2.10.10. Creating security group and roles in AWS
  2.10.11. CloudFormation template for security objects
  2.10.12. RHCOS AMIs for the AWS infrastructure
    2.10.11.1. AWS regions without a published RHCOS AMI
    2.10.11.2. Uploading a custom RHCOS AMI in AWS
  2.10.12. Creating the bootstrap node in AWS
    2.10.12.1. CloudFormation template for the bootstrap machine
  2.10.13. Creating the control plane machines in AWS
    2.10.13.1. CloudFormation template for control plane machines
  2.10.14. Creating the worker nodes in AWS
    2.10.14.1. CloudFormation template for worker machines
  2.10.15. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure
  2.10.16. Installing the OpenShift CLI by downloading the binary
    2.10.16.1. Installing the OpenShift CLI on Linux
    2.10.16.2. Installing the OpenShift CLI on Windows
    2.10.16.3. Installing the OpenShift CLI on macOS
  2.10.17. Logging in to the cluster by using the CLI
  2.10.18. Approving the certificate signing requests for your machines
2.10.19. Initial Operator configuration
   2.10.19.1. Image registry storage configuration
      2.10.19.1.1. Configuring registry storage for AWS with user-provisioned infrastructure
      2.10.19.1.2. Configuring storage for the image registry in non-production clusters
   2.10.20. Deleting the bootstrap resources
   2.10.21. Creating the Ingress DNS Records
   2.10.22. Completing an AWS installation on user-provisioned infrastructure
   2.10.23. Logging in to the cluster by using the web console
   2.10.24. Telemetry access for OpenShift Container Platform
   2.10.25. Additional resources
   2.10.26. Next steps

2.11. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE
   2.11.1. Prerequisites
   2.11.2. About installations in restricted networks
      2.11.2.1. Additional limits
   2.11.3. Internet access for OpenShift Container Platform
   2.11.4. Required AWS infrastructure components
      2.11.4.1. Cluster machines
      2.11.4.2. Other infrastructure components
      2.11.4.3. Certificate signing requests management
      2.11.4.4. Required AWS permissions
   2.11.5. Generating an SSH private key and adding it to the agent
   2.11.6. Creating the installation files for AWS
      2.11.6.1. Optional: Creating a separate /var partition
      2.11.6.2. Creating the installation configuration file
      2.11.6.3. Configuring the cluster-wide proxy during installation
      2.11.6.4. Creating the Kubernetes manifest and Ignition config files
   2.11.7. Extracting the infrastructure name
   2.11.8. Creating a VPC in AWS
      2.11.8.1. CloudFormation template for the VPC
   2.11.9. Creating networking and load balancing components in AWS
      2.11.9.1. CloudFormation template for the network and load balancers
   2.11.10. Creating security group and roles in AWS
      2.11.10.1. CloudFormation template for security objects
   2.11.11. RHCOS AMIs for the AWS infrastructure
   2.11.12. Creating the bootstrap node in AWS
      2.11.12.1. CloudFormation template for the bootstrap machine
   2.11.13. Creating the control plane machines in AWS
      2.11.13.1. CloudFormation template for control plane machines
   2.11.14. Creating the worker nodes in AWS
      2.11.14.1. CloudFormation template for worker machines
   2.11.15. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure
   2.11.16. Logging in to the cluster by using the CLI
   2.11.17. Approving the certificate signing requests for your machines
   2.11.18. Initial Operator configuration
      2.11.18.1. Disabling the default OperatorHub sources
      2.11.18.2. Image registry storage configuration
         2.11.18.2.1. Configuring registry storage for AWS with user-provisioned infrastructure
         2.11.18.2.2. Configuring storage for the image registry in non-production clusters
   2.11.19. Deleting the bootstrap resources
   2.11.20. Creating the Ingress DNS Records
   2.11.21. Completing an AWS installation on user-provisioned infrastructure
2.11.2. Logging in to the cluster by using the web console
2.11.3. Telemetry access for OpenShift Container Platform
2.11.4. Additional resources
2.11.5. Next steps
2.12. UNINSTALLING A CLUSTER ON AWS
    2.12.1. Removing a cluster that uses installer-provisioned infrastructure

CHAPTER 3. INSTALLING ON AZURE

3.1. CONFIGURING AN AZURE ACCOUNT
    3.1.1. Azure account limits
    3.1.2. Configuring a public DNS zone in Azure
    3.1.3. Increasing Azure account limits
    3.1.4. Required Azure roles
    3.1.5. Creating a service principal
    3.1.6. Supported Azure regions
        Supported Azure public regions
        Supported Azure Government regions
    3.1.7. Next steps
3.2. MANUALLY CREATING IAM FOR AZURE
    3.2.1. Alternatives to storing administrator-level secrets in the kube-system project
    3.2.2. Manually create IAM
    3.2.3. Admin credentials root secret format
    3.2.4. Upgrading clusters with manually maintained credentials
    3.2.5. Mint mode
    3.2.6. Next steps
3.3. INSTALLING A CLUSTER QUICKLY ON AZURE
    3.3.1. Prerequisites
    3.3.2. Internet access for OpenShift Container Platform
    3.3.3. Generating an SSH private key and adding it to the agent
    3.3.4. Obtaining the installation program
    3.3.5. Deploying the cluster
    3.3.6. Installing the OpenShift CLI by downloading the binary
        3.3.6.1. Installing the OpenShift CLI on Linux
        3.3.6.2. Installing the OpenShift CLI on Windows
        3.3.6.3. Installing the OpenShift CLI on macOS
    3.3.7. Logging in to the cluster by using the CLI
    3.3.8. Telemetry access for OpenShift Container Platform
    3.3.9. Next steps
3.4. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS
    3.4.1. Prerequisites
    3.4.2. Internet access for OpenShift Container Platform
    3.4.3. Generating an SSH private key and adding it to the agent
    3.4.4. Obtaining the installation program
    3.4.5. Creating the installation configuration file
        3.4.5.1. Installation configuration parameters
            3.4.5.1.1. Required configuration parameters
            3.4.5.1.2. Network configuration parameters
            3.4.5.1.3. Optional configuration parameters
            3.4.5.1.4. Additional Azure configuration parameters
        3.4.5.2. Sample customized install-config.yaml file for Azure
        3.4.5.3. Configuring the cluster-wide proxy during installation
    3.4.6. Deploying the cluster
    3.4.7. Installing the OpenShift CLI by downloading the binary
3.4.7.1. Installing the OpenShift CLI on Linux
3.4.7.2. Installing the OpenShift CLI on Windows
3.4.7.3. Installing the OpenShift CLI on macOS
3.4.8. Logging in to the cluster by using the CLI
3.4.9. Telemetry access for OpenShift Container Platform
3.4.10. Next steps

3.5. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS
3.5.1. Prerequisites
3.5.2. Internet access for OpenShift Container Platform
3.5.3. Generating an SSH private key and adding it to the agent
3.5.4. Obtaining the installation program
3.5.5. Creating the installation configuration file
   3.5.5.1. Installation configuration parameters
      3.5.5.1.1. Required configuration parameters
      3.5.5.1.2. Network configuration parameters
      3.5.5.1.3. Optional configuration parameters
      3.5.5.1.4. Additional Azure configuration parameters
      3.5.5.2. Sample customized install-config.yaml file for Azure
      3.5.5.3. Configuring the cluster-wide proxy during installation
3.5.6. Network configuration phases
3.5.7. Specifying advanced network configuration
3.5.8. Cluster Network Operator configuration
   3.5.8.1. Cluster Network Operator configuration object
      defaultNetwork object configuration
      Configuration for the OpenShift SDN CNI cluster network provider
      Configuration for the OVN-Kubernetes CNI cluster network provider
      kubeProxyConfig object configuration
3.5.9. Configuring hybrid networking with OVN-Kubernetes
3.5.10. Deploying the cluster
3.5.11. Installing the OpenShift CLI by downloading the binary
   3.5.11.1. Installing the OpenShift CLI on Linux
   3.5.11.2. Installing the OpenShift CLI on Windows
   3.5.11.3. Installing the OpenShift CLI on macOS
3.5.12. Logging in to the cluster by using the CLI
3.5.13. Telemetry access for OpenShift Container Platform
3.5.14. Next steps

3.6. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET
3.6.1. Prerequisites
3.6.2. About reusing a VNet for your OpenShift Container Platform cluster
   3.6.2.1. Requirements for using your VNet
      3.6.2.1.1. Network security group requirements
   3.6.2.2. Division of permissions
   3.6.2.3. Isolation between clusters
3.6.3. Internet access for OpenShift Container Platform
3.6.4. Generating an SSH private key and adding it to the agent
3.6.5. Obtaining the installation program
3.6.6. Creating the installation configuration file
   3.6.6.1. Installation configuration parameters
      3.6.6.1.1. Required configuration parameters
      3.6.6.1.2. Network configuration parameters
      3.6.6.1.3. Optional configuration parameters
      3.6.6.1.4. Additional Azure configuration parameters
      3.6.6.2. Sample customized install-config.yaml file for Azure
3.6.6.3. Configuring the cluster-wide proxy during installation
3.6.7. Deploying the cluster
3.6.8. Installing the OpenShift CLI by downloading the binary
  3.6.8.1. Installing the OpenShift CLI on Linux
  3.6.8.2. Installing the OpenShift CLI on Windows
  3.6.8.3. Installing the OpenShift CLI on macOS
3.6.9. Logging in to the cluster by using the CLI
3.6.10. Telemetry access for OpenShift Container Platform
3.6.11. Next steps

3.7. INSTALLING A PRIVATE CLUSTER ON AZURE
3.7.1. Prerequisites
3.7.2. Private clusters
  3.7.2.1. Private clusters in Azure
    3.7.2.1.1. Limitations
  3.7.2.2. User-defined outbound routing
    Private cluster with network address translation
    Private cluster with Azure Firewall
    Private cluster with a proxy configuration
    Private cluster with no Internet access
3.7.3. About reusing a VNet for your OpenShift Container Platform cluster
  3.7.3.1. Requirements for using your VNet
    3.7.3.1.1. Network security group requirements
  3.7.3.2. Division of permissions
  3.7.3.3. Isolation between clusters
3.7.4. Internet access for OpenShift Container Platform
3.7.5. Generating an SSH private key and adding it to the agent
3.7.6. Obtaining the installation program
3.7.7. Manually creating the installation configuration file
  3.7.7.1. Installation configuration parameters
    3.7.7.1.1. Required configuration parameters
    3.7.7.1.2. Network configuration parameters
    3.7.7.1.3. Optional configuration parameters
    3.7.7.1.4. Additional Azure configuration parameters
    3.7.7.2. Sample customized install-config.yaml file for Azure
    3.7.7.3. Configuring the cluster-wide proxy during installation
3.7.8. Deploying the cluster
3.7.9. Installing the OpenShift CLI by downloading the binary
  3.7.9.1. Installing the OpenShift CLI on Linux
  3.7.9.2. Installing the OpenShift CLI on Windows
  3.7.9.3. Installing the OpenShift CLI on macOS
3.7.10. Logging in to the cluster by using the CLI
3.7.11. Telemetry access for OpenShift Container Platform
3.7.12. Next steps

3.8. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION
3.8.1. Prerequisites
3.8.2. Azure government regions
3.8.3. Private clusters
  3.8.3.1. Private clusters in Azure
    3.8.3.1.1. Limitations
  3.8.3.2. User-defined outbound routing
    Private cluster with network address translation
    Private cluster with Azure Firewall
    Private cluster with a proxy configuration
Private cluster with no Internet access

3.8.4. About reusing a VNet for your OpenShift Container Platform cluster
  3.8.4.1. Requirements for using your VNet
    3.8.4.1.1. Network security group requirements
    3.8.4.2. Division of permissions
    3.8.4.3. Isolation between clusters

3.8.5. Internet access for OpenShift Container Platform

3.8.6. Generating an SSH private key and adding it to the agent

3.8.7. Obtaining the installation program

3.8.8. Manually creating the installation configuration file
  3.8.8.1. Installation configuration parameters
    3.8.8.1.1. Required configuration parameters
    3.8.8.1.2. Network configuration parameters
    3.8.8.1.3. Optional configuration parameters
    3.8.8.1.4. Additional Azure configuration parameters
  3.8.8.2. Sample customized install-config.yaml file for Azure
  3.8.8.3. Configuring the cluster-wide proxy during installation

3.8.9. Deploying the cluster

3.8.10. Installing the OpenShift CLI by downloading the binary
  3.8.10.1. Installing the OpenShift CLI on Linux
  3.8.10.2. Installing the OpenShift CLI on Windows
  3.8.10.3. Installing the OpenShift CLI on macOS

3.8.11. Logging in to the cluster by using the CLI

3.8.12. Telemetry access for OpenShift Container Platform

3.8.13. Next steps

3.9. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES

3.9.1. Prerequisites

3.9.2. Internet access for OpenShift Container Platform

3.9.3. Configuring your Azure project
  3.9.3.1. Azure account limits
  3.9.3.2. Configuring a public DNS zone in Azure
  3.9.3.3. Increasing Azure account limits
  3.9.3.4. Certificate signing requests management
  3.9.3.5. Required Azure roles
  3.9.3.6. Creating a service principal
  3.9.3.7. Supported Azure regions
    Supported Azure public regions
    Supported Azure Government regions

3.9.4. Obtaining the installation program

3.9.5. Generating an SSH private key and adding it to the agent

3.9.6. Creating the installation files for Azure
  3.9.6.1. Optional: Creating a separate /var partition
  3.9.6.2. Creating the installation configuration file
  3.9.6.3. Configuring the cluster-wide proxy during installation
  3.9.6.4. Exporting common variables for ARM templates
  3.9.6.5. Creating the Kubernetes manifest and Ignition config files

3.9.7. Creating the Azure resource group and identity

3.9.8. Uploading the RHCOS cluster image and bootstrap Ignition config file

3.9.9. Example for creating DNS zones

3.9.10. Creating a VNet in Azure
  3.9.10.1. ARM template for the VNet
  3.9.11. Deploying the RHCOS cluster image for the Azure infrastructure
    3.9.11.1. ARM template for image storage
3.9.12. Networking requirements for user-provisioned infrastructure
   Network topology requirements
   Load balancers
3.9.13. Creating networking and load balancing components in Azure
   3.9.13.1. ARM template for the network and load balancers
   3.9.14.1. ARM template for the bootstrap machine
3.9.15. Creating the control plane machines in Azure
   3.9.15.1. ARM template for control plane machines
3.9.16. Wait for bootstrap completion and remove bootstrap resources in Azure
3.9.17. Creating additional worker machines in Azure
   3.9.17.1. ARM template for worker machines
3.9.18. Installing the OpenShift CLI by downloading the binary
   3.9.18.1. Installing the OpenShift CLI on Linux
   3.9.18.2. Installing the OpenShift CLI on Windows
   3.9.18.3. Installing the OpenShift CLI on macOS
3.9.19. Logging into the cluster by using the CLI
3.9.20. Approving the certificate signing requests for your machines
3.9.21. Adding the Ingress DNS records
3.9.22. Completing an Azure installation on user-provisioned infrastructure
3.9.23. Telemetry access for OpenShift Container Platform

3.10. UNINSTALLING A CLUSTER ON AZURE
3.10.1. Removing a cluster that uses installer-provisioned infrastructure

CHAPTER 4. INSTALLING ON GCP
4.1. CONFIGURING A GCP PROJECT
   4.1.1. Creating a GCP project
   4.1.2. Enabling API services in GCP
   4.1.3. Configuring DNS for GCP
   4.1.4. GCP account limits
   4.1.5. Creating a service account in GCP
      4.1.5.1. Required GCP permissions
   4.1.6. Supported GCP regions
   4.1.7. Next steps
4.2. MANUALLY CREATING IAM FOR GCP
   4.2.1. Alternatives to storing administrator-level secrets in the kube-system project
   4.2.2. Manually create IAM
   4.2.3. Admin credentials root secret format
   4.2.4. Upgrading clusters with manually maintained credentials
   4.2.5. Mint mode
   4.2.6. Mint Mode with removal or rotation of the admin credential
   4.2.7. Next steps
4.3. INSTALLING A CLUSTER QUICKLY ON GCP
   4.3.1. Prerequisites
   4.3.2. Internet access for OpenShift Container Platform
   4.3.3. Generating an SSH private key and adding it to the agent
   4.3.4. Obtaining the installation program
   4.3.5. Deploying the cluster
   4.3.6. Installing the OpenShift CLI by downloading the binary
      4.3.6.1. Installing the OpenShift CLI on Linux
      4.3.6.2. Installing the OpenShift CLI on Windows
      4.3.6.3. Installing the OpenShift CLI on macOS
   4.3.7. Logging into the cluster by using the CLI
4.3.8. Telemetry access for OpenShift Container Platform
4.3.9. Next steps

4.4. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS
4.4.1. Prerequisites
4.4.2. Internet access for OpenShift Container Platform
4.4.3. Generating an SSH private key and adding it to the agent
4.4.4. Obtaining the installation program
4.4.5. Creating the installation configuration file
  4.4.5.1. Installation configuration parameters
    4.4.5.1.1. Required configuration parameters
    4.4.5.1.2. Network configuration parameters
    4.4.5.1.3. Optional configuration parameters
    4.4.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters
  4.4.5.2. Sample customized install-config.yaml file for GCP
  4.4.5.3. Configuring the cluster-wide proxy during installation
4.4.6. Deploying the cluster
4.4.7. Installing the OpenShift CLI by downloading the binary
  4.4.7.1. Installing the OpenShift CLI on Linux
  4.4.7.2. Installing the OpenShift CLI on Windows
  4.4.7.3. Installing the OpenShift CLI on macOS
4.4.8. Logging in to the cluster by using the CLI
4.4.9. Telemetry access for OpenShift Container Platform
4.4.10. Next steps

4.5. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS
4.5.1. Prerequisites
4.5.2. Internet access for OpenShift Container Platform
4.5.3. Generating an SSH private key and adding it to the agent
4.5.4. Obtaining the installation program
4.5.5. Creating the installation configuration file
  4.5.5.1. Installation configuration parameters
    4.5.5.1.1. Required configuration parameters
    4.5.5.1.2. Network configuration parameters
    4.5.5.1.3. Optional configuration parameters
    4.5.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters
  4.5.5.2. Sample customized install-config.yaml file for GCP
  4.5.5.3. Configuring the cluster-wide proxy during installation
4.5.6. Network configuration phases
4.5.7. Specifying advanced network configuration
4.5.8. Cluster Network Operator configuration
  4.5.8.1. Cluster Network Operator configuration object
    defaultNetwork object configuration
    Configuration for the OpenShift SDN CNI cluster network provider
    Configuration for the OVN-Kubernetes CNI cluster network provider
    kubeProxyConfig object configuration
4.5.9. Deploying the cluster
4.5.10. Installing the OpenShift CLI by downloading the binary
  4.5.10.1. Installing the OpenShift CLI on Linux
  4.5.10.2. Installing the OpenShift CLI on Windows
  4.5.10.3. Installing the OpenShift CLI on macOS
4.5.11. Logging in to the cluster by using the CLI
4.5.12. Telemetry access for OpenShift Container Platform
4.5.13. Next steps

4.6. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK
4.6.1. Prerequisites

4.6.2. About installations in restricted networks

4.6.2.1. Additional limits

4.6.3. Internet access for OpenShift Container Platform

4.6.4. Generating an SSH private key and adding it to the agent

4.6.5. Creating the installation configuration file

4.6.5.1. Installation configuration parameters

4.6.5.1.1. Required configuration parameters

4.6.5.1.2. Network configuration parameters

4.6.5.1.3. Optional configuration parameters

4.6.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

4.6.5.2. Sample customized install-config.yaml file for GCP

4.6.5.3. Configuring the cluster-wide proxy during installation

4.6.6. Deploying the cluster

4.6.7. Installing the OpenShift CLI by downloading the binary

4.6.7.1. Installing the OpenShift CLI on Linux

4.6.7.2. Installing the OpenShift CLI on Windows

4.6.7.3. Installing the OpenShift CLI on macOS

4.6.8. Logging in to the cluster by using the CLI

4.6.9. Disabling the default OperatorHub sources

4.6.10. Telemetry access for OpenShift Container Platform

4.6.11. Next steps

4.7. INSTALLING A CLUSTER ON GCP INTO AN EXISTING VPC

4.7.1. Prerequisites

4.7.2. Internet access for OpenShift Container Platform

4.7.3. Generating an SSH private key and adding it to the agent

4.7.4. Obtaining the installation program

4.7.5. Creating the installation configuration file

4.7.5.1. Installation configuration parameters

4.7.5.1.1. Required configuration parameters

4.7.5.1.2. Network configuration parameters

4.7.5.1.3. Optional configuration parameters

4.7.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

4.7.5.2. Sample customized install-config.yaml file for GCP

4.7.5.3. Configuring the cluster-wide proxy during installation

4.7.6. Deploying the cluster

4.7.7. Installing the OpenShift CLI by downloading the binary

4.7.7.1. Installing the OpenShift CLI on Linux

4.7.7.2. Installing the OpenShift CLI on Windows

4.7.7.3. Installing the OpenShift CLI on macOS

4.7.8. Logging in to the cluster by using the CLI

4.7.9. Telemetry access for OpenShift Container Platform

4.7.10. Next steps

4.8. INSTALLING A PRIVATE CLUSTER ON GCP

4.8.1. Prerequisites

4.8.2. Private clusters

4.8.2.1. Private clusters in GCP

4.8.2.1.1. Limitations

4.8.3. About using a custom VPC

4.8.3.1. Requirements for using your VPC

4.8.3.2. Division of permissions

4.8.3.3. Isolation between clusters

4.8.4. Internet access for OpenShift Container Platform
4.8.5. Generating an SSH private key and adding it to the agent
4.8.6. Obtaining the installation program
4.8.7. Manually creating the installation configuration file
  4.8.7.1. Installation configuration parameters
    4.8.7.1.1. Required configuration parameters
    4.8.7.1.2. Network configuration parameters
    4.8.7.1.3. Optional configuration parameters
    4.8.7.1.4. Additional Google Cloud Platform (GCP) configuration parameters
  4.8.7.2. Sample customized install-config.yaml file for GCP
  4.8.7.3. Configuring the cluster-wide proxy during installation
4.8.8. Deploying the cluster
4.8.9. Installing the OpenShift CLI by downloading the binary
  4.8.9.1. Installing the OpenShift CLI on Linux
  4.8.9.2. Installing the OpenShift CLI on Windows
  4.8.9.3. Installing the OpenShift CLI on macOS
4.8.10. Logging in to the cluster by using the CLI
4.8.11. Telemetry access for OpenShift Container Platform
4.8.12. Next steps

4.9. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN GCP BY USING DEPLOYMENT MANAGER TEMPLATES
4.9.1. Prerequisites
4.9.2. Certificate signing requests management
4.9.3. Internet access for OpenShift Container Platform
4.9.4. Configuring your GCP project
  4.9.4.1. Creating a GCP project
  4.9.4.2. Enabling API services in GCP
  4.9.4.3. Configuring DNS for GCP
  4.9.4.4. GCP account limits
  4.9.4.5. Creating a service account in GCP
    4.9.4.5.1. Required GCP permissions
  4.9.4.6. Supported GCP regions
  4.9.4.7. Installing and configuring CLI tools for GCP
4.9.5. Creating the installation files for GCP
  4.9.5.1. Optional: Creating a separate /var partition
  4.9.5.2. Creating the installation configuration file
  4.9.5.3. Configuring the cluster-wide proxy during installation
  4.9.5.4. Creating the Kubernetes manifest and Ignition config files
4.9.6. Exporting common variables
  4.9.6.1. Extracting the infrastructure name
  4.9.6.2. Exporting common variables for Deployment Manager templates
4.9.7. Creating a VPC in GCP
  4.9.7.1. Deployment Manager template for the VPC
4.9.8. Networking requirements for user-provisioned infrastructure
  Network topology requirements
  Load balancers
4.9.9. Creating load balancers in GCP
  4.9.9.1. Deployment Manager template for the external load balancer
  4.9.9.2. Deployment Manager template for the internal load balancer
4.9.10. Creating a private DNS zone in GCP
  4.9.10.1. Deployment Manager template for the private DNS
4.9.11. Creating firewall rules in GCP
  4.9.11.1. Deployment Manager template for firewall rules
4.9.12. Creating IAM roles in GCP
4.9.12.1. Deployment Manager template for IAM roles
4.9.13. Creating the RHcos cluster image for the GCP infrastructure
4.9.14. Creating the bootstrap machine in GCP
4.9.14.1. Deployment Manager template for the bootstrap machine
4.9.15. Creating the control plane machines in GCP
4.9.15.1. Deployment Manager template for control plane machines
4.9.16. Wait for bootstrap completion and remove bootstrap resources in GCP
4.9.17. Creating additional worker machines in GCP
4.9.17.1. Deployment Manager template for worker machines
4.9.18. Installing the OpenShift CLI by downloading the binary
4.9.18.1. Installing the OpenShift CLI on Linux
4.9.18.2. Installing the OpenShift CLI on Windows
4.9.18.3. Installing the OpenShift CLI on macOS
4.9.19. Logging in to the cluster by using the CLI
4.9.20. Approving the certificate signing requests for your machines
4.9.21. Optional: Adding the ingress DNS records
4.9.22. Completing a GCP installation on user-provisioned infrastructure
4.9.23. Telemetry access for OpenShift Container Platform
4.9.24. Next steps

4.10. INSTALLING A CLUSTER INTO A SHARED VPC ON GCP USING DEPLOYMENT MANAGER TEMPLATES

4.10.1. Prerequisites
4.10.2. Certificate signing requests management
4.10.3. Internet access for OpenShift Container Platform
4.10.4. Configuring the GCP project that hosts your cluster
4.10.4.1. Creating a GCP project
4.10.4.2. Enabling API services in GCP
4.10.4.3. GCP account limits
4.10.4.4. Creating a service account in GCP
4.10.4.4.1. Required GCP permissions
4.10.4.5. Supported GCP regions
4.10.4.6. Installing and configuring CLI tools for GCP
4.10.5. Configuring the GCP project that hosts your shared VPC network
4.10.5.1. Configuring DNS for GCP
4.10.5.2. Creating a VPC in GCP
4.10.5.2.1. Deployment Manager template for the VPC
4.10.6. Creating the installation files for GCP
4.10.6.1. Manually creating the installation configuration file
4.10.6.2. Sample customized install-config.yaml file for GCP
4.10.6.3. Configuring the cluster-wide proxy during installation
4.10.6.4. Creating the Kubernetes manifest and Ignition config files
4.10.7. Exporting common variables
4.10.7.1. Extracting the infrastructure name
4.10.7.2. Exporting common variables for Deployment Manager templates
4.10.8. Networking requirements for user-provisioned infrastructure
4.10.9. Creating load balancers in GCP
4.10.9.1. Deployment Manager template for the external load balancer
4.10.9.2. Deployment Manager template for the internal load balancer
4.10.10. Creating a private DNS zone in GCP
4.10.10.1. Deployment Manager template for the private DNS
4.10.11. Creating firewall rules in GCP
4.10.11.1. Deployment Manager template for firewall rules
4.10.12. Creating IAM roles in GCP
4.10.12.1. Deployment Manager template for IAM roles
4.10.13. Creating the RHCOS cluster image for the GCP infrastructure
4.10.14. Creating the bootstrap machine in GCP
4.10.14.1. Deployment Manager template for the bootstrap machine
4.10.15. Creating the control plane machines in GCP
4.10.15.1. Deployment Manager template for control plane machines
4.10.16. Wait for bootstrap completion and remove bootstrap resources in GCP
4.10.17. Creating additional worker machines in GCP
4.10.17.1. Deployment Manager template for worker machines
4.10.18. Installing the OpenShift CLI by downloading the binary
4.10.18.1. Installing the OpenShift CLI on Linux
4.10.18.2. Installing the OpenShift CLI on Windows
4.10.18.3. Installing the OpenShift CLI on macOS
4.10.19. Logging in to the cluster by using the CLI
4.10.20. Approving the certificate signing requests for your machines
4.10.21. Adding the ingress DNS records
4.10.22. Adding ingress firewall rules
4.10.22.1. Creating cluster-wide firewall rules for a shared VPC in GCP
4.10.23. Completing a GCP installation on user-provisioned infrastructure
4.10.24. Telemetry access for OpenShift Container Platform
4.10.25. Next steps

4.11. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

4.11.1. Prerequisites
4.11.2. About installations in restricted networks
4.11.2.1. Additional limits
4.11.3. Internet access for OpenShift Container Platform
4.11.4. Configuring your GCP project
4.11.4.1. Creating a GCP project
4.11.4.2. Enabling API services in GCP
4.11.4.3. Configuring DNS for GCP
4.11.4.4. GCP account limits
4.11.4.5. Creating a service account in GCP
4.11.4.5.1. Required GCP permissions
4.11.4.6. Supported GCP regions
4.11.4.7. Installing and configuring CLI tools for GCP
4.11.5. Creating the installation files for GCP
4.11.5.1. Optional: Creating a separate /var partition
4.11.5.2. Creating the installation configuration file
4.11.5.3. Configuring the cluster-wide proxy during installation
4.11.5.4. Creating the Kubernetes manifest and Ignition config files
4.11.6. Exporting common variables
4.11.6.1. Extracting the infrastructure name
4.11.6.2. Exporting common variables for Deployment Manager templates
4.11.7. Creating a VPC in GCP
4.11.7.1. Deployment Manager template for the VPC
4.11.8. Networking requirements for user-provisioned infrastructure
   Network topology requirements
   Load balancers
4.11.9. Creating load balancers in GCP
4.11.9.1. Deployment Manager template for the external load balancer
4.11.9.2. Deployment Manager template for the internal load balancer
4.11.10. Creating a private DNS zone in GCP
   4.11.10.1. Deployment Manager template for the private DNS
4.11.11. Creating firewall rules in GCP
   4.11.11.1. Deployment Manager template for firewall rules
4.11.12. Creating IAM roles in GCP
   4.11.12.1. Deployment Manager template for IAM roles
4.11.13. Creating the RHCOs cluster image for the GCP infrastructure
4.11.14. Creating the bootstrap machine in GCP
   4.11.14.1. Deployment Manager template for the bootstrap machine
4.11.15. Creating the control plane machines in GCP
   4.11.15.1. Deployment Manager template for control plane machines
4.11.16. Wait for bootstrap completion and remove bootstrap resources in GCP
4.11.17. Creating additional worker machines in GCP
   4.11.17.1. Deployment Manager template for worker machines
4.11.18. Logging in to the cluster by using the CLI
4.11.19. Disabling the default OperatorHub sources
4.11.20. Approving the certificate signing requests for your machines
4.11.21. Optional: Adding the ingress DNS records
4.11.22. Completing a GCP installation on user-provisioned infrastructure
4.11.23. Telemetry access for OpenShift Container Platform
4.11.24. Next steps
4.12. UNINSTALLING A CLUSTER ON GCP
   4.12.1. Removing a cluster that uses installer-provisioned infrastructure

CHAPTER 5. INSTALLING ON BARE METAL
5.1. INSTALLING A CLUSTER ON BARE METAL
   5.1.1. Prerequisites
   5.1.2. Internet access for OpenShift Container Platform
   5.1.3. Machine requirements for a cluster with user-provisioned infrastructure
      5.1.3.1. Required machines
      5.1.3.2. Network connectivity requirements
      5.1.3.3. Minimum resource requirements
      5.1.3.4. Certificate signing requests management
   5.1.4. Creating the user-provisioned infrastructure
      5.1.4.1. Networking requirements for user-provisioned infrastructure
      Network topology requirements
      Load balancers
      NTP configuration
      5.1.4.2. User-provisioned DNS requirements
   5.1.5. Generating an SSH private key and adding it to the agent
   5.1.6. Obtaining the installation program
   5.1.7. Installing the OpenShift CLI by downloading the binary
      5.1.7.1. Installing the OpenShift CLI on Linux
      5.1.7.2. Installing the OpenShift CLI on Windows
      5.1.7.3. Installing the OpenShift CLI on macOS
   5.1.8. Manually creating the installation configuration file
      5.1.8.1. Installation configuration parameters
      5.1.8.1.1. Required configuration parameters
      5.1.8.1.2. Network configuration parameters
      5.1.8.1.3. Optional configuration parameters
      5.1.8.2. Sample install-config.yaml file for bare metal
      5.1.8.3. Configuring the cluster-wide proxy during installation
5.1.9. Configuring a three-node cluster
5.1.10. Creating the Kubernetes manifest and Ignition config files
5.1.11. Installing RH COS and starting the OpenShift Container Platform bootstrap process
  5.1.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
  5.1.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
  5.1.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration
    5.1.11.3.1. Using advanced networking options for PXE and ISO installations
    5.1.11.3.2. Disk partitioning
      5.1.11.3.2.1. Creating a separate /var partition
      5.1.11.3.2.2. Retaining existing partitions
    5.1.11.3.3. Identifying Ignition configs
      5.1.11.3.3.1. Embedding an Ignition config in the RHCOS ISO
    5.1.11.3.4. Advanced RHCOS installation reference
    coreos.inst boot options for ISO or PXE install
    coreos-installer options for ISO install
  5.1.11.4. Routing and bonding options at RHCOS boot prompt
  5.1.11.5. Image registry removed during installation
  5.1.11.6. Image registry storage configuration
    5.1.11.6.1. Configuring registry storage for bare metal and other manual installations
    5.1.11.6.2. Configuring storage for the image registry in non-production clusters
    5.1.11.6.3. Configuring block registry storage
  5.1.11.7. Completing installation on user-provisioned infrastructure
  5.1.11.8. Telemetry access for OpenShift Container Platform
5.1.12. Creating the cluster
5.1.13. Logging in to the cluster by using the CLI
5.1.14. Approving the certificate signing requests for your machines
5.1.15. Initial Operator configuration
  5.1.15.1. Image registry removed during installation
  5.1.15.2. Image registry storage configuration
    5.1.15.2.1. Configuring registry storage for bare metal and other manual installations
  5.1.15.2.2. Configuring storage for the image registry in non-production clusters
  5.1.15.2.3. Configuring block registry storage
  5.1.16. Completing installation on user-provisioned infrastructure
  5.1.17. Telemetry access for OpenShift Container Platform
5.1.18. Next steps

5.2. INSTALLING A CLUSTER ON BARE METAL WITH NETWORK CUSTOMIZATIONS
5.2.1. Prerequisites
5.2.2. Internet access for OpenShift Container Platform
5.2.3. Machine requirements for a cluster with user-provisioned infrastructure
  5.2.3.1. Required machines
  5.2.3.2. Network connectivity requirements
  5.2.3.3. Minimum resource requirements
  5.2.3.4. Certificate signing requests management
5.2.4. Creating the user-provisioned infrastructure
  5.2.4.1. Networking requirements for user-provisioned infrastructure
    Network topology requirements
    Load balancers
    NTP configuration
  5.2.4.2. User-provisioned DNS requirements
5.2.5. Generating an SSH private key and adding it to the agent
5.2.6. Obtaining the installation program
5.2.7. Installing the OpenShift CLI by downloading the binary
  5.2.7.1. Installing the OpenShift CLI on Linux
  5.2.7.2. Installing the OpenShift CLI on Windows
  5.2.7.3. Installing the OpenShift CLI on macOS
5.2.8. Manually creating the installation configuration file
  5.2.8.1. Installation configuration parameters
    5.2.8.1.1. Required configuration parameters
    5.2.8.1.2. Network configuration parameters
    5.2.8.1.3. Optional configuration parameters
5.2.8.2. Sample install-config.yaml file for bare metal
5.2.9. Network configuration phases
5.2.10. Specifying advanced network configuration
5.2.11. Cluster Network Operator configuration
  5.2.11.1. Cluster Network Operator configuration object
    defaultNetwork object configuration
    Configuration for the OpenShift SDN CNI cluster network provider
    Configuration for the OVN-Kubernetes CNI cluster network provider
    kubeProxyConfig object configuration
5.2.12. Creating the Ignition config files
5.2.13. Installing RHCOS and starting the OpenShift Container Platform bootstrap process
  5.2.13.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
  5.2.13.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
  5.2.13.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration
    5.2.13.3.1. Using advanced networking options for PXE and ISO installations
    5.2.13.3.2. Disk partitioning
      5.2.13.3.2.1. Creating a separate /var partition
      5.2.13.3.2.2. Retaining existing partitions
    5.2.13.3.3. Identifying Ignition configs
      5.2.13.3.3.1. Embedding an Ignition config in the RHCOS ISO
    5.2.13.3.4. Advanced RHCOS installation reference
    coreos.inst boot options for ISO or PXE install
    coreos-installer options for ISO install
5.2.14. Creating the cluster
5.2.15. Logging in to the cluster by using the CLI
5.2.16. Approving the certificate signing requests for your machines
5.2.17. Initial Operator configuration
  5.2.17.1. Image registry removed during installation
  5.2.17.2. Image registry storage configuration
  5.2.17.3. Configuring block registry storage
5.2.18. Completing installation on user-provisioned infrastructure
  5.2.18.1. Networking requirements for user-provisioned infrastructure
  Network topology requirements
  Load balancers
  NTP configuration
  5.2.18.2. User-provisioned DNS requirements
5.2.19. Telemetry access for OpenShift Container Platform
5.2.20. Next steps
5.3. INSTALLING A CLUSTER ON BARE METAL IN A RESTRICTED NETWORK
  5.3.1. Prerequisites
  5.3.2. About installations in restricted networks
    5.3.2.1. Additional limits
  5.3.3. Internet access for OpenShift Container Platform
  5.3.4. Machine requirements for a cluster with user-provisioned infrastructure
    5.3.4.1. Required machines
    5.3.4.2. Network connectivity requirements
    5.3.4.3. Minimum resource requirements
    5.3.4.4. Certificate signing requests management
  5.3.5. Creating the user-provisioned infrastructure
    5.3.5.1. Networking requirements for user-provisioned infrastructure
      Network topology requirements
      Load balancers
      NTP configuration
    5.3.5.2. User-provisioned DNS requirements
  5.3.6. Generating an SSH private key and adding it to the agent
  5.3.7. Manually creating the installation configuration file
    5.3.7.1. Installation configuration parameters
5.3.7.1. Required configuration parameters
5.3.7.2. Sample install-config.yaml file for bare metal
5.3.7.3. Configuring the cluster-wide proxy during installation
5.3.8. Configuring a three-node cluster
5.3.9. Creating the Kubernetes manifest and Ignition config files
5.3.10. Configuring chrony time service
5.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process
5.3.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
5.3.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
5.3.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration
5.3.11.3.1. Using advanced networking options for PXE and ISO installations
5.3.11.3.2. Disk partitioning
5.3.11.3.2.1. Creating a separate /var partition
5.3.11.3.2.2. Retaining existing partitions
5.3.11.3.3. Identifying Ignition configs
5.3.11.3.3.1. Embedding an Ignition config in the RHCOS ISO
5.3.11.3.4. Advanced RHCOS installation reference
Routing and bonding options at RHCOS boot prompt
coreos.inst boot options for ISO or PXE install
coreos-installer options for ISO install
5.3.12. Creating the cluster
5.3.13. Logging in to the cluster by using the CLI
5.3.14. Approving the certificate signing requests for your machines
5.3.15. Initial Operator configuration
5.3.15.1. Disabling the default OperatorHub sources
5.3.15.2. Image registry storage configuration
5.3.15.2.1. Changing the image registry's management state
5.3.15.2.2. Configuring registry storage for bare metal and other manual installations
5.3.15.2.3. Configuring storage for the image registry in non-production clusters
5.3.15.2.4. Configuring block registry storage
5.3.16. Completing installation on user-provisioned infrastructure
5.3.17. Telemetry access for OpenShift Container Platform
5.3.18. Next steps

CHAPTER 6. DEPLOYING INSTALLER-PROVISIONED CLUSTERS ON BARE METAL

6.1. OVERVIEW
6.2. PREREQUISITES
6.2.1. Node requirements
6.2.2. Network requirements
6.2.3. Configuring nodes
6.2.4. Out-of-band management
6.2.5. Required data for installation
6.2.6. Validation checklist for nodes
6.3. SETTING UP THE ENVIRONMENT FOR AN OPENSIGNED INSTALLATION
6.3.1. Installing RHEL on the provisioner node
6.3.2. Preparing the provisioner node for OpenShift Container Platform installation
6.3.3. Retrieving the OpenShift Container Platform installer
6.3.4. Extracting the OpenShift Container Platform installer
6.3.5. Creating an RHCOS images cache (optional)
6.3.6. Configuration files
6.3.6.1. Configuring the install-config.yaml file
6.3.6.2. Setting proxy settings within the install-config.yaml file (optional)
6.3.6.3. Modifying the install-config.yaml file for no provisioning network (optional)
6.3.6.4. Additional install-config parameters
6.3.6.5. BMC addressing
6.3.6.6. Root device hints
6.3.6.7. Creating the OpenShift Container Platform manifests
6.3.7. Creating a disconnected registry (optional)
   6.3.7.1. Preparing the registry node to host the mirrored registry (optional)
   6.3.7.2. Generating the self-signed certificate (optional)
   6.3.7.3. Creating the registry podman container (optional)
   6.3.7.4. Copy and update the pull-secret (optional)
   6.3.7.5. Mirroring the repository (optional)
   6.3.7.6. Modify the install-config.yaml file to use the disconnected registry (optional)
6.3.8. Deploying routers on worker nodes
6.3.9. Validation checklist for installation
6.3.10. Deploying the cluster via the OpenShift Container Platform installer
6.3.11. Following the installation
6.4. EXPANDING THE CLUSTER
   6.4.1. Preparing the bare metal node
   6.4.2. Provisioning the bare metal node
6.5. TROUBLESHOOTING
   6.5.1. Troubleshooting the installer workflow
   6.5.2. Troubleshooting install-config.yaml
   6.5.3. Bootstrap VM issues
      6.5.3.1. Bootstrap VM cannot boot up the cluster nodes
      6.5.3.2. Inspecting logs
   6.5.4. Cluster nodes will not PXE boot
   6.5.5. The API is not accessible
   6.5.6. Cleaning up previous installations
   6.5.7. Issues with creating the registry
   6.5.8. Miscellaneous issues
      6.5.8.1. Addressing the runtime network not ready error
      6.5.8.2. Cluster nodes not getting the correct IPv6 address over DHCP
      6.5.8.3. Cluster nodes not getting the correct hostname over DHCP
      6.5.8.4. Routes do not reach endpoints
      6.5.8.5. Failed Ignition during Firstboot
      6.5.8.6. NTP out of sync
   6.5.9. Reviewing the installation

CHAPTER 7. INSTALLING ON IBM Z AND LINUXONE

7.1. INSTALLING A CLUSTER ON IBM Z AND LINUXONE
   7.1.1. Prerequisites
   7.1.2. Internet access for OpenShift Container Platform
   7.1.3. Machine requirements for a cluster with user-provisioned infrastructure
      7.1.3.1. Required machines
      7.1.3.2. Network connectivity requirements
      7.1.3.3. IBM Z network connectivity requirements
      7.1.3.4. Minimum resource requirements
      7.1.3.5. Minimum IBM Z system environment
         Hardware requirements
         Operating system requirements
         IBM Z network connectivity requirements
         Disk storage for the z/VM guest virtual machines
Storage / Main Memory
7.1.3.6. Preferred IBM Z system environment
    Hardware requirements
    Operating system requirements
    IBM Z network connectivity requirements
    Disk storage for the z/VM guest virtual machines
7.1.3.7. Certificate signing requests management
7.1.4. Creating the user-provisioned infrastructure
    7.1.4.1. Networking requirements for user-provisioned infrastructure
        Network topology requirements
        Load balancers
        NTP configuration
    7.1.4.2. User-provisioned DNS requirements
7.1.5. Generating an SSH private key and adding it to the agent
7.1.6. Obtaining the installation program
7.1.7. Installing the OpenShift CLI by downloading the binary
    7.1.7.1. Installing the OpenShift CLI on Linux
    7.1.7.2. Installing the OpenShift CLI on Windows
    7.1.7.3. Installing the OpenShift CLI on macOS
7.1.8. Manually creating the installation configuration file
    7.1.8.1. Installation configuration parameters
        7.1.8.1.1. Required configuration parameters
        7.1.8.1.2. Network configuration parameters
        7.1.8.1.3. Optional configuration parameters
        7.1.8.2. Sample install-config.yaml file for IBM Z
    7.1.9. Configuring the cluster-wide proxy during installation
7.1.10. Creating the Kubernetes manifest and Ignition config files
7.1.11. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines
        7.1.11.1. Advanced RHCOS installation reference
            Routing and bonding options at RHCOS boot prompt
7.1.12. Creating the cluster
7.1.13. Logging in to the cluster by using the CLI
7.1.14. Approving the certificate signing requests for your machines
7.1.15. Initial Operator configuration
        7.1.15.1. Image registry storage configuration
        7.1.15.1.1. Configuring registry storage for IBM Z
        7.1.15.1.2. Configuring storage for the image registry in non-production clusters
7.1.16. Completing installation on user-provisioned infrastructure
7.1.17. Telemetry access for OpenShift Container Platform
7.1.18. Collecting debugging information
7.1.19. Next steps

7.2. INSTALLING A CLUSTER ON IBM Z AND LINUXONE IN A RESTRICTED NETWORK
7.2.1. About installations in restricted networks
    7.2.1.1. Additional limits
7.2.2. Internet access for OpenShift Container Platform
7.2.3. Machine requirements for a cluster with user-provisioned infrastructure
        7.2.3.1. Required machines
        7.2.3.2. Network connectivity requirements
        7.2.3.3. IBM Z network connectivity requirements
        7.2.3.4. Minimum resource requirements
        7.2.3.5. Minimum IBM Z system environment
            Hardware requirements
Operating system requirements 1263
IBM Z network connectivity requirements 1263
Disk storage for the z/VM guest virtual machines 1263
Storage / Main Memory 1264
7.2.3.6. Preferred IBM Z system environment 1264
Hardware requirements 1264
Operating system requirements 1264
IBM Z network connectivity requirements 1264
Disk storage for the z/VM guest virtual machines 1264
Storage / Main Memory 1265
7.2.3.7. Certificate signing requests management 1265
7.2.4. Creating the user-provisioned infrastructure 1265
7.2.4.1. Networking requirements for user-provisioned infrastructure 1265
Network topology requirements 1267
Load balancers 1267
NTP configuration 1268
7.2.4.2. User-provisioned DNS requirements 1269
7.2.5. Generating an SSH private key and adding it to the agent 1271
7.2.6. Manually creating the installation configuration file 1273
7.2.6.1. Installation configuration parameters 1274
7.2.6.1. Required configuration parameters 1274
7.2.6.1.2. Network configuration parameters 1275
7.2.6.1.3. Optional configuration parameters 1277
7.2.6.2. Sample install-config.yaml file for IBM Z 1282
7.2.6.3. Configuring the cluster-wide proxy during installation 1285
7.2.7. Creating the Kubernetes manifest and Ignition config files 1286
7.2.8. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines 1287
7.2.8.1. Advanced RHCOS installation reference 1289
Routing and bonding options at RHCOS boot prompt 1289
7.2.9. Creating the cluster 1292
7.2.10. Logging in to the cluster by using the CLI 1293
7.2.11. Approving the certificate signing requests for your machines 1294
7.2.12. Initial Operator configuration 1296
7.2.12.1. Disabling the default OperatorHub sources 1297
7.2.12.2. Image registry storage configuration 1298
7.2.12.1. Configuring registry storage for IBM Z 1298
7.2.12.2.2. Configuring storage for the image registry in non-production clusters 1299
7.2.13. Completing installation on user-provisioned infrastructure 1300
7.2.14. Telemetry access for OpenShift Container Platform 1302
7.2.15. Collecting debugging information 1302
7.2.16. Next steps 1303

CHAPTER 8. INSTALLING ON IBM POWER SYSTEMS 1304
8.1. INSTALLING A CLUSTER ON IBM POWER SYSTEMS 1304
8.1.1. Internet access for OpenShift Container Platform 1304
8.1.2. Machine requirements for a cluster with user-provisioned infrastructure 1305
8.1.2.1. Required machines 1305
8.1.2.2. Network connectivity requirements 1305
8.1.2.3. Minimum resource requirements 1305
8.1.2.4. Certificate signing requests management 1306
8.1.3. Creating the user-provisioned infrastructure 1306
8.1.3.1. Networking requirements for user-provisioned infrastructure 1306
Network topology requirements 1308
Load balancers
NTP configuration
8.1.3.2. User-provisioned DNS requirements
8.1.4. Generating an SSH private key and adding it to the agent
8.1.5. Obtaining the installation program
8.1.6. Installing the OpenShift CLI by downloading the binary
  8.1.6.1. Installing the OpenShift CLI on Linux
  8.1.6.2. Installing the OpenShift CLI on Windows
  8.1.6.3. Installing the OpenShift CLI on macOS
8.1.7. Manually creating the installation configuration file
  8.1.7.1. Sample install-config.yaml file for IBM Power Systems
  8.1.7.2. Configuring the cluster-wide proxy during installation
8.1.8. Creating the Kubernetes manifest and Ignition config files
8.1.9. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines
  8.1.9.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
    Routing and bonding options at RHCOS boot prompt
  8.1.9.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
8.1.10. Creating the cluster
8.1.11. Logging in to the cluster by using the CLI
8.1.12. Approving the certificate signing requests for your machines
8.1.13. Initial Operator configuration
  8.1.13.1. Image registry storage configuration
    8.1.13.1.1. Configuring registry storage for IBM Power Systems
    8.1.13.1.2. Configuring storage for the image registry in non-production clusters
  8.1.14. Completing installation on user-provisioned infrastructure
8.1.15. Telemetry access for OpenShift Container Platform
8.1.16. Next steps
8.2. INSTALLING A CLUSTER ON IBM POWER SYSTEMS IN A RESTRICTED NETWORK
8.2.1. About installations in restricted networks
  8.2.1.1. Additional limits
8.2.2. Internet access for OpenShift Container Platform
8.2.3. Machine requirements for a cluster with user-provisioned infrastructure
  8.2.3.1. Required machines
  8.2.3.2. Network connectivity requirements
  8.2.3.3. Minimum resource requirements
  8.2.3.4. Certificate signing requests management
8.2.4. Creating the user-provisioned infrastructure
  8.2.4.1. Networking requirements for user-provisioned infrastructure
    Network topology requirements
    Load balancers
    NTP configuration
  8.2.4.2. User-provisioned DNS requirements
8.2.5. Generating an SSH private key and adding it to the agent
8.2.6. Manually creating the installation configuration file
  8.2.6.1. Sample install-config.yaml file for IBM Power Systems
  8.2.6.2. Configuring the cluster-wide proxy during installation
8.2.7. Creating the Kubernetes manifest and Ignition config files
8.2.8. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines
  8.2.8.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
    Routing and bonding options at RHCOS boot prompt
  8.2.8.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
8.2.9. Creating the cluster
8.2.10. Logging in to the cluster by using the CLI
8.2.11. Approving the certificate signing requests for your machines
8.2.12. Initial Operator configuration
  8.2.12.1. Disabling the default OperatorHub sources
  8.2.12.2. Image registry storage configuration
  8.2.12.2.1. Changing the image registry’s management state
  8.2.12.2.2. Configuring registry storage for IBM Power Systems
  8.2.12.2.3. Configuring storage for the image registry in non-production clusters
8.2.13. Completing installation on user-provisioned infrastructure
8.2.14. Telemetry access for OpenShift Container Platform
8.2.15. Next steps

CHAPTER 9. INSTALLING ON OPENSTACK ................................................. 1376
9.1. INSTALLING A CLUSTER ON OPENSTACK WITH CUSTOMIZATIONS 1376
  9.1.1. Prerequisites
  9.1.2. Resource guidelines for installing OpenShift Container Platform on RHOSP
    9.1.2.1. Control plane and compute machines
    9.1.2.2. Bootstrap machine
  9.1.3. Internet access for OpenShift Container Platform
  9.1.4. Enabling Swift on RHOSP
  9.1.5. Verifying external network access
  9.1.6. Defining parameters for the installation program
  9.1.7. Obtaining the installation program
  9.1.8. Creating the installation configuration file
    9.1.8.1. Configuring the cluster-wide proxy during installation
  9.1.9. Installation configuration parameters
    9.1.9.1. Required configuration parameters
    9.1.9.2. Network configuration parameters
    9.1.9.3. Optional configuration parameters
    9.1.9.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters
    9.1.9.5. Optional RHOSP configuration parameters
    9.1.9.6. Custom subnets in RHOSP deployments
    9.1.9.7. Sample customized install-config.yaml file for RHOSP
  9.1.10. Setting compute machine affinity
  9.1.11. Generating an SSH private key and adding it to the agent
  9.1.12. Enabling access to the environment
    9.1.12.1. Enabling access with floating IP addresses
    9.1.12.2. Completing installation without floating IP addresses
  9.1.13. Deploying the cluster
  9.1.14. Verifying cluster status
  9.1.15. Logging in to the cluster by using the CLI
  9.1.16. Telemetry access for OpenShift Container Platform
  9.1.17. Next steps

9.2. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR 1406
  9.2.1. Prerequisites
  9.2.2. About Kuryr SDN
  9.2.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr
    9.2.3.1. Increasing quota
    9.2.3.2. Configuring Neutron
    9.2.3.3. Configuring Octavia
      9.2.3.3.1. The Octavia OVN Driver
    9.2.3.4. Known limitations of installing with Kuryr

Table of Contents
RHOSP general limitations
RHOSP version limitations
RHOSP environment limitations
RHOSP upgrade limitations
9.2.3.5. Control plane and compute machines
9.2.3.6. Bootstrap machine
9.2.4. Internet access for OpenShift Container Platform
9.2.5. Enabling Swift on RHOSP
9.2.6. Verifying external network access
9.2.7. Defining parameters for the installation program
9.2.8. Obtaining the installation program
9.2.9. Creating the installation configuration file
9.2.9.1. Configuring the cluster-wide proxy during installation
9.2.10. Installation configuration parameters
9.2.10.1. Required configuration parameters
9.2.10.2. Network configuration parameters
9.2.10.3. Optional configuration parameters
9.2.10.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters
9.2.10.5. Optional RHOSP configuration parameters
9.2.10.6. Custom subnets in RHOSP deployments
9.2.10.7. Sample customized install-config.yaml file for RHOSP with Kuryr
9.2.10.8. Kuryr ports pools
9.2.10.9. Adjusting Kuryr ports pools during installation
9.2.11. Setting compute machine affinity
9.2.12. Generating an SSH private key and adding it to the agent
9.2.13. Enabling access to the environment
9.2.13.1. Enabling access with floating IP addresses
9.2.13.2. Completing installation without floating IP addresses
9.2.14. Deploying the cluster
9.2.15. Verifying cluster status
9.2.16. Logging in to the cluster by using the CLI
9.2.17. Telemetry access for OpenShift Container Platform
9.2.18. Next steps

9.3. INSTALLING A CLUSTER ON OPENSTACK ON YOUR OWN INFRASTRUCTURE
9.3.1. Prerequisites
9.3.2. Internet access for OpenShift Container Platform
9.3.3. Resource guidelines for installing OpenShift Container Platform on RHOSP
9.3.3.1. Control plane and compute machines
9.3.3.2. Bootstrap machine
9.3.4. Downloading playbook dependencies
9.3.5. Downloading the installation playbooks
9.3.6. Obtaining the installation program
9.3.7. Generating an SSH private key and adding it to the agent
9.3.8. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image
9.3.9. Verifying external network access
9.3.10. Enabling access to the environment
9.3.10.1. Enabling access with floating IP addresses
9.3.10.2. Completing installation without floating IP addresses
9.3.11. Defining parameters for the installation program
9.3.12. Creating the installation configuration file
9.3.13. Installation configuration parameters
9.3.13.1. Required configuration parameters
9.3.13.2. Network configuration parameters
9.3.13.3. Optional configuration parameters
9.3.13.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters
9.3.13.5. Optional RHOSP configuration parameters
9.3.13.6. Custom subnets in RHOSP deployments
9.3.13.7. Sample customized install-config.yaml file for RHOSP
9.3.13.8. Setting a custom subnet for machines
9.3.13.9. Emptying compute machine pools

9.3.14. Creating the Kubernetes manifest and Ignition config files
9.3.15. Preparing the bootstrap Ignition files
9.3.16. Creating control plane Ignition config files on RHOSP
9.3.17. Creating network resources on RHOSP
9.3.18. Creating the bootstrap machine on RHOSP
9.3.19. Creating the control plane machines on RHOSP
9.3.20. Logging into the cluster by using the CLI
9.3.21. Deleting bootstrap resources from RHOSP
9.3.22. Creating compute machines on RHOSP
9.3.23. Approving the certificate signing requests for your machines
9.3.24. Verifying a successful installation
9.3.25. Telemetry access for OpenShift Container Platform
9.3.26. Next steps

9.4. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR ON YOUR OWN INFRASTRUCTURE
9.4.1. Prerequisites
9.4.2. About Kuryr SDN
9.4.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr
  9.4.3.1. Increasing quota
  9.4.3.2. Configuring Neutron
  9.4.3.3. Configuring Octavia
    9.4.3.3.1. The Octavia OVN Driver
  9.4.3.4. Known limitations of installing with Kuryr
    RHOSP general limitations
    RHOSP version limitations
    RHOSP environment limitations
    RHOSP upgrade limitations
  9.4.3.5. Control plane and compute machines
  9.4.3.6. Bootstrap machine
9.4.4. Internet access for OpenShift Container Platform
9.4.5. Downloading playbook dependencies
9.4.6. Downloading the installation playbooks
9.4.7. Obtaining the installation program
9.4.8. Generating an SSH private key and adding it to the agent
9.4.9. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image
9.4.10. Verifying external network access
9.4.11. Enabling access to the environment
  9.4.11.1. Enabling access with floating IP addresses
  9.4.11.2. Completing installation without floating IP addresses
9.4.12. Defining parameters for the installation program
9.4.13. Creating the installation configuration file
  9.4.14.1. Required configuration parameters
  9.4.14.2. Network configuration parameters
  9.4.14.3. Optional configuration parameters
  9.4.14.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters
  9.4.14.5. Optional RHOSP configuration parameters
9.4.14.6. Custom subnets in RHOSP deployments
9.4.14.7. Sample customized install-config.yaml file for RHOSP with Kuryr
9.4.14.9. Adjusting Kuryr ports pools during installation
9.4.14.10. Setting a custom subnet for machines
9.4.14.11. Emptying compute machine pools
9.4.14.12. Modifying the network type
9.4.15. Creating the Kubernetes manifest and Ignition config files
9.4.16. Preparing the bootstrap Ignition files
9.4.17. Creating control plane Ignition config files on RHOSP
9.4.18. Creating network resources on RHOSP
9.4.19. Creating the bootstrap machine on RHOSP
9.4.20. Creating the control plane machines on RHOSP
9.4.21. Logging in to the cluster by using the CLI
9.4.22. Deleting bootstrap resources from RHOSP
9.4.23. Creating compute machines on RHOSP
9.4.24. Approving the certificate signing requests for your machines
9.4.25. Verifying a successful installation
9.4.26. Telemetry access for OpenShift Container Platform
9.4.27. Next steps

9.5. INSTALLING A CLUSTER ON OPENSTACK IN A RESTRICTED NETWORK
9.5.1. About installations in restricted networks
9.5.1.1. Additional limits
9.5.2. Resource guidelines for installing OpenShift Container Platform on RHOSP
9.5.2.1. Control plane and compute machines
9.5.2.2. Bootstrap machine
9.5.3. Internet access for OpenShift Container Platform
9.5.4. Enabling Swift on RHOSP
9.5.5. Defining parameters for the installation program
9.5.6. Creating the RHCOS image for restricted network installations
9.5.7. Creating the installation configuration file
9.5.7.1. Configuring the cluster-wide proxy during installation
9.5.7.2. Installation configuration parameters
9.5.7.2.1. Required configuration parameters
9.5.7.2.2. Network configuration parameters
9.5.7.2.3. Optional configuration parameters
9.5.7.2.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters
9.5.7.2.5. Optional RHOSP configuration parameters
9.5.7.3. Sample customized install-config.yaml file for restricted OpenStack installations
9.5.8. Setting compute machine affinity
9.5.9. Generating an SSH private key and adding it to the agent
9.5.10. Enabling access to the environment
9.5.10.1. Enabling access with floating IP addresses
9.5.10.2. Completing installation without floating IP addresses
9.5.11. Deploying the cluster
9.5.12. Verifying cluster status
9.5.13. Logging in to the cluster by using the CLI
9.5.14. Disabling the default OperatorHub sources
9.5.15. Telemetry access for OpenShift Container Platform
9.5.16. Next steps

9.6. UNINSTALLING A CLUSTER ON OPENSTACK
9.6.1. Removing a cluster that uses installer-provisioned infrastructure

9.7. UNINSTALLING A CLUSTER ON RHOSP FROM YOUR OWN INFRASTRUCTURE
9.7.1. Downloading playbook dependencies 1572
9.7.2. Removing a cluster from RHOSP that uses your own infrastructure 1573

CHAPTER 10. INSTALLING ON RHV ................................. 1574

10.1. INSTALLING A CLUSTER QUICKLY ON RHV .................. 1574
10.1.1. Prerequisites 1574
10.1.2. Internet access for OpenShift Container Platform 1575
10.1.3. Requirements for the RHV environment 1575
10.1.4. Verifying the requirements for the RHV environment 1576
10.1.5. Preparing the network environment on RHV 1578
10.1.6. Setting up the CA certificate for RHV 1579
10.1.7. Generating an SSH private key and adding it to the agent 1580
10.1.8. Obtaining the installation program 1581
10.1.9. Deploying the cluster 1582
10.1.10. Installing the OpenShift CLI by downloading the binary 1585
  10.1.10.1. Installing the OpenShift CLI on Linux 1585
  10.1.10.2. Installing the OpenShift CLI on Windows 1586
  10.1.10.3. Installing the OpenShift CLI on macOS 1586
10.1.11. Logging in to the cluster by using the CLI 1587
10.1.12. Verifying cluster status 1587
10.1.13. Accessing the OpenShift Container Platform web console on RHV 1588
10.1.14. Telemetry access for OpenShift Container Platform 1589
10.1.15. Troubleshooting common issues with installing on Red Hat Virtualization (RHV) 1589
  10.1.15.1. CPU load increases and nodes go into a Not Ready state 1589
  10.1.15.2. Trouble connecting the OpenShift Container Platform cluster API 1589
10.1.16. Post-installation tasks 1590

10.2. INSTALLING A CLUSTER ON RHV WITH CUSTOMIZATIONS 1590
10.2.1. Prerequisites 1591
10.2.2. Internet access for OpenShift Container Platform 1591
10.2.3. Requirements for the RHV environment 1591
10.2.4. Verifying the requirements for the RHV environment 1593
10.2.5. Preparing the network environment on RHV 1595
10.2.6. Setting up the CA certificate for RHV 1596
10.2.7. Generating an SSH private key and adding it to the agent 1596
10.2.8. Obtaining the installation program 1598
10.2.9. Creating the installation configuration file 1599
  10.2.9.1. Example install-config.yaml files for Red Hat Virtualization (RHV) 1601
  10.2.9.2. Installation configuration parameters 1603
    10.2.9.2.1. Required configuration parameters 1603
    10.2.9.2.2. Network configuration parameters 1605
    10.2.9.2.3. Optional configuration parameters 1606
    10.2.9.2.4. Additional Red Hat Virtualization (RHV) configuration parameters 1610
    10.2.9.2.5. Additional RHV parameters for machine pools 1611
10.2.10. Deploying the cluster 1612
10.2.11. Installing the OpenShift CLI by downloading the binary 1614
  10.2.11.1. Installing the OpenShift CLI on Linux 1614
  10.2.11.2. Installing the OpenShift CLI on Windows 1614
  10.2.11.3. Installing the OpenShift CLI on macOS 1615
10.2.12. Logging in to the cluster by using the CLI 1615
10.2.13. Verifying cluster status 1616
10.2.14. Accessing the OpenShift Container Platform web console on RHV 1617
10.2.15. Telemetry access for OpenShift Container Platform 1617
10.2.16. Troubleshooting common issues with installing on Red Hat Virtualization (RHV) 1617
10.2.16.1. CPU load increases and nodes go into a Not Ready state
10.2.16.2. Trouble connecting the OpenShift Container Platform cluster API
10.2.17. Post-installation tasks
10.2.18. Next steps

10.3. INSTALLING A CLUSTER ON RHV WITH USER-PROVISIONED INFRASTRUCTURE

10.3.1. Prerequisites
10.3.2. Internet access for OpenShift Container Platform
10.3.3. Requirements for the RHV environment
10.3.4. Verifying the requirements for the RHV environment
10.3.5. Networking requirements for user-provisioned infrastructure
   Network topology requirements
   Load balancers
   NTP configuration
10.3.6. Setting up the installation machine
10.3.7. Setting up the CA certificate for RHV
10.3.8. Generating an SSH private key and adding it to the agent
10.3.9. Obtaining the installation program
10.3.10. Downloading the Ansible playbooks
10.3.11. The inventory.yml file
10.3.12. Specifying the RHCOS image settings
10.3.13. Creating the install config file
10.3.14. Customizing install-config.yaml
10.3.15. Generate manifest files
10.3.16. Making control-plane nodes non-schedulable
10.3.17. Building the Ignition files
10.3.18. Creating templates and virtual machines
10.3.19. Creating the bootstrap machine
10.3.20. Creating the control plane nodes
10.3.21. Verifying cluster status
10.3.22. Removing the bootstrap machine
10.3.23. Creating the worker nodes and completing the installation
10.3.24. Telemetry access for OpenShift Container Platform

10.4. UNINSTALLING A CLUSTER ON RHV

10.4.1. Removing a cluster that uses installer-provisioned infrastructure
10.4.2. Removing a cluster that uses user-provisioned infrastructure

CHAPTER 11. INSTALLING ON VSPHERE

11.1. INSTALLING A CLUSTER ON VSPHERE

11.1.1. Prerequisites
11.1.2. Internet access for OpenShift Container Platform
11.1.3. VMware vSphere infrastructure requirements
11.1.4. vCenter requirements
   Required vCenter account privileges
   Using OpenShift Container Platform with vMotion
   Cluster resources
   Cluster limits
   Networking requirements
   Required IP Addresses
   DNS records
11.1.5. Generating an SSH private key and adding it to the agent
11.1.6. Obtaining the installation program
11.1.7. Adding vCenter root CA certificates to your system trust
11.1.8. Deploying the cluster
11.1.9. Installing the OpenShift CLI by downloading the binary
  11.1.9.1. Installing the OpenShift CLI on Linux
  11.1.9.2. Installing the OpenShift CLI on Windows
  11.1.9.3. Installing the OpenShift CLI on macOS
11.1.10. Logging in to the cluster by using the CLI
11.1.11. Creating registry storage
  11.1.11.1. Image registry removed during installation
  11.1.11.2. Image registry storage configuration
    11.1.11.2.1. Configuring registry storage for VMware vSphere
    11.1.11.2.2. Configuring block registry storage for VMware vSphere
  11.1.12. Backing up VMware vSphere volumes
11.1.13. Telemetry access for OpenShift Container Platform
11.1.14. Next steps
11.2. INSTALLING A CLUSTER ON VSPHERE WITH CUSTOMIZATIONS
  11.2.1. Prerequisites
  11.2.2. Internet access for OpenShift Container Platform
  11.2.3. VMware vSphere infrastructure requirements
  11.2.4. vCenter requirements
    Required vCenter account privileges
    Using OpenShift Container Platform with vMotion
    Cluster resources
    Cluster limits
    Networking requirements
      Required IP Addresses
      DNS records
  11.2.5. Generating an SSH private key and adding it to the agent
  11.2.6. Obtaining the installation program
  11.2.7. Adding vCenter root CA certificates to your system trust
  11.2.8. Creating the installation configuration file
    11.2.8.1. Installation configuration parameters
      11.2.8.1.1. Required configuration parameters
      11.2.8.1.2. Network configuration parameters
      11.2.8.1.3. Optional configuration parameters
      11.2.8.1.4. Additional VMware vSphere configuration parameters
      11.2.8.1.5. Optional VMware vSphere machine pool configuration parameters
    11.2.8.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster
    11.2.8.3. Configuring the cluster-wide proxy during installation
  11.2.9. Deploying the cluster
  11.2.10. Installing the OpenShift CLI by downloading the binary
    11.2.10.1. Installing the OpenShift CLI on Linux
    11.2.10.2. Installing the OpenShift CLI on Windows
    11.2.10.3. Installing the OpenShift CLI on macOS
  11.2.11. Logging in to the cluster by using the CLI
  11.2.12. Creating registry storage
    11.2.12.1. Image registry removed during installation
    11.2.12.2. Image registry storage configuration
      11.2.12.2.1. Configuring registry storage for VMware vSphere
      11.2.12.2.2. Configuring block registry storage for VMware vSphere
  11.2.13. Backing up VMware vSphere volumes
  11.2.14. Telemetry access for OpenShift Container Platform
  11.2.15. Next steps
11.3. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS
  11.3.1. Prerequisites
11.3.2. Internet access for OpenShift Container Platform
11.3.3. VMware vSphere infrastructure requirements
11.3.4. vCenter requirements
   Required vCenter account privileges
   Using OpenShift Container Platform with vMotion
   Cluster resources
   Cluster limits
   Networking requirements
   Required IP Addresses
   DNS records
11.3.5. Generating an SSH private key and adding it to the agent
11.3.6. Obtaining the installation program
11.3.7. Adding vCenter root CA certificates to your system trust
11.3.8. Creating the installation configuration file
   11.3.8.1. Installation configuration parameters
      11.3.8.1.1. Required configuration parameters
      11.3.8.1.2. Network configuration parameters
      11.3.8.1.3. Optional configuration parameters
      11.3.8.1.4. Additional VMware vSphere configuration parameters
      11.3.8.1.5. Optional VMware vSphere machine pool configuration parameters
   11.3.8.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster
   11.3.8.3. Configuring the cluster-wide proxy during installation
11.3.9. Network configuration phases
11.3.10. Specifying advanced network configuration
11.3.11. Cluster Network Operator configuration
   11.3.11.1. Cluster Network Operator configuration object
      defaultNetwork object configuration
      Configuration for the OpenShift SDN CNI cluster network provider
      Configuration for the OVN-Kubernetes CNI cluster network provider
      kubeProxyConfig object configuration
11.3.12. Deploying the cluster
11.3.13. Installing the OpenShift CLI by downloading the binary
   11.3.13.1. Installing the OpenShift CLI on Linux
   11.3.13.2. Installing the OpenShift CLI on Windows
   11.3.13.3. Installing the OpenShift CLI on macOS
11.3.14. Logging in to the cluster by using the CLI
11.3.15. Creating registry storage
   11.3.15.1. Image registry removed during installation
   11.3.15.2. Image registry storage configuration
      11.3.15.2.1. Configuring registry storage for VMware vSphere
      11.3.15.2.2. Configuring block registry storage for VMware vSphere
11.3.16. Backing up VMware vSphere volumes
11.3.17. Telemetry access for OpenShift Container Platform
11.3.18. Next steps
11.4. INSTALLING A CLUSTER ON VSPHERE WITH USER-PROVISIONED INFRASTRUCTURE
11.4.1. Prerequisites
11.4.2. Internet access for OpenShift Container Platform
11.4.3. VMware vSphere infrastructure requirements
11.4.4. Machine requirements for a cluster with user-provisioned infrastructure
   11.4.4.1. Required machines
   11.4.4.2. Network connectivity requirements
   11.4.4.3. Minimum resource requirements
   11.4.4.4. Certificate signing requests management
11.4.5. Creating the user-provisioned infrastructure

11.4.5.1. Networking requirements for user-provisioned infrastructure
Network topology requirements
Load balancers
 Ethernet adaptor hardware address requirements
NTP configuration

11.4.5.2. User-provisioned DNS requirements

11.4.6. Generating an SSH private key and adding it to the agent

11.4.7. Obtaining the installation program

11.4.8. Manually creating the installation configuration file
11.4.8.1. Sample install-config.yaml file for VMware vSphere
11.4.8.2. Configuring the cluster-wide proxy during installation

11.4.9. Creating the Kubernetes manifest and Ignition config files

11.4.10. Extracting the infrastructure name

11.4.11. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

11.4.12. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

11.4.13. Disk partitioning
Creating a separate /var partition

11.4.14. Installing the OpenShift CLI by downloading the binary
11.4.14.1. Installing the OpenShift CLI on Linux
11.4.14.2. Installing the OpenShift CLI on Windows
11.4.14.3. Installing the OpenShift CLI on macOS

11.4.15. Creating the cluster

11.4.16. Logging in to the cluster by using the CLI

11.4.17. Approving the certificate signing requests for your machines

11.4.18. Initial Operator configuration
11.4.18.1. Image registry removed during installation
11.4.18.2. Image registry storage configuration
11.4.18.2.1. Configuring registry storage for VMware vSphere
11.4.18.2.2. Configuring storage for the image registry in non-production clusters
11.4.18.2.3. Configuring block registry storage for VMware vSphere

11.4.19. Completing installation on user-provisioned infrastructure

11.4.20. Backing up VMware vSphere volumes

11.4.21. Telemetry access for OpenShift Container Platform

11.4.22. Next steps

11.5. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS

11.5.1. Prerequisites

11.5.2. Internet access for OpenShift Container Platform

11.5.3. VMware vSphere infrastructure requirements

11.5.4. Machine requirements for a cluster with user-provisioned infrastructure
11.5.4.1. Required machines
11.5.4.2. Network connectivity requirements
11.5.4.3. Minimum resource requirements
11.5.4.4. Certificate signing requests management

11.5.5. Creating the user-provisioned infrastructure
11.5.5.1. Networking requirements for user-provisioned infrastructure
Network topology requirements
Load balancers
 Ethernet adaptor hardware address requirements
NTP configuration

11.5.5.2. User-provisioned DNS requirements

11.5.6. Generating an SSH private key and adding it to the agent

11.5.7. Obtaining the installation program
11.5.8. Manually creating the installation configuration file
   11.5.8.1. Sample install-config.yaml file for VMware vSphere
   11.5.8.2. Configuring the cluster-wide proxy during installation
11.5.9. Network configuration phases
11.5.10. Specifying advanced network configuration
11.5.11. Cluster Network Operator configuration
   11.5.11.1. Cluster Network Operator configuration object
   - defaultNetwork object configuration
   - Configuration for the OpenShift SDN CNI cluster network provider
   - Configuration for the OVN-Kubernetes CNI cluster network provider
   - kubeProxyConfig object configuration
11.5.12. Creating the Ignition config files
11.5.13. Extracting the infrastructure name
11.5.14. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
11.5.15. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
11.5.16. Disk partitioning
   - Creating a separate /var partition
11.5.17. Creating the cluster
11.5.18. Logging in to the cluster by using the CLI
11.5.19. Approving the certificate signing requests for your machines
   11.5.19.1. Initial Operator configuration
   11.5.19.2. Image registry removed during installation
   11.5.19.3. Image registry storage configuration
      11.5.19.3.1. Configuring block registry storage for VMware vSphere
11.5.20. Completing installation on user-provisioned infrastructure
11.5.21. Backing up VMware vSphere volumes
11.5.22. Telemetry access for OpenShift Container Platform
11.5.23. Next steps

11.6. INSTALLING A CLUSTER ON VSphere IN A RESTRICTED NETWORK
11.6.1. Prerequisites
11.6.2. About installations in restricted networks
   11.6.2.1. Additional limits
11.6.3. Internet access for OpenShift Container Platform
11.6.4. VMware vSphere infrastructure requirements
11.6.5. vCenter requirements
   - Required vCenter account privileges
   - Using OpenShift Container Platform with vMotion
   - Cluster resources
   - Cluster limits
   - Networking requirements
   - Required IP Addresses
   - DNS records
11.6.6. Generating an SSH private key and adding it to the agent
11.6.7. Adding vCenter root CA certificates to your system trust
11.6.8. Creating the RHCOS image for restricted network installations
11.6.9. Creating the installation configuration file
   11.6.9.1. Installation configuration parameters
      11.6.9.1.1. Required configuration parameters
      11.6.9.1.2. Network configuration parameters
      11.6.9.1.3. Optional configuration parameters
      11.6.9.1.4. Additional VMware vSphere configuration parameters
      11.6.9.1.5. Optional VMware vSphere machine pool configuration parameters
   11.6.9.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster
11.6.9.3. Configuring the cluster-wide proxy during installation

11.6.10. Deploying the cluster

11.6.11. Installing the OpenShift CLI by downloading the binary
   11.6.11.1. Installing the OpenShift CLI on Linux
   11.6.11.2. Installing the OpenShift CLI on Windows
   11.6.11.3. Installing the OpenShift CLI on macOS

11.6.12. Logging in to the cluster by using the CLI

11.6.13. Disabling the default OperatorHub sources

11.6.14. Creating registry storage
   11.6.14.1. Image registry removed during installation
   11.6.14.2. Image registry storage configuration
      11.6.14.2.1. Configuring registry storage for VMware vSphere

11.6.15. Telemetry access for OpenShift Container Platform

11.6.16. Next steps

11.7. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

11.7.1. Prerequisites

11.7.2. About installations in restricted networks
   11.7.2.1. Additional limits

11.7.3. Internet access for OpenShift Container Platform

11.7.4. VMware vSphere infrastructure requirements

11.7.5. Machine requirements for a cluster with user-provisioned infrastructure
   11.7.5.1. Required machines
   11.7.5.2. Network connectivity requirements
   11.7.5.3. Minimum resource requirements
   11.7.5.4. Certificate signing requests management

11.7.6. Creating the user-provisioned infrastructure
   11.7.6.1. Networking requirements for user-provisioned infrastructure
      Network topology requirements
      Load balancers
      Ethernet adaptor hardware address requirements
      NTP configuration
   11.7.6.2. User-provisioned DNS requirements

11.7.7. Generating an SSH private key and adding it to the agent

11.7.8. Manually creating the installation configuration file
   11.7.8.1. Sample install-config.yaml file for VMware vSphere
   11.7.8.2. Configuring the cluster-wide proxy during installation

11.7.9. Creating the Kubernetes manifest and Ignition config files

11.7.10. Configuring chrony time service

11.7.11. Extracting the infrastructure name

11.7.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

11.7.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

11.7.14. Disk partitioning
      Creating a separate /var partition

11.7.15. Creating the cluster

11.7.16. Logging in to the cluster by using the CLI

11.7.17. Approving the certificate signing requests for your machines

11.7.18. Initial Operator configuration
   11.7.18.1. Disabling the default OperatorHub sources
   11.7.18.2. Image registry storage configuration
      11.7.18.2.1. Configuring registry storage for VMware vSphere
      11.7.18.2.2. Configuring storage for the image registry in non-production clusters
      11.7.18.2.3. Configuring block registry storage for VMware vSphere
11.7.19. Completing installation on user-provisioned infrastructure
11.7.20. Backing up VMware vSphere volumes
11.7.21. Telemetry access for OpenShift Container Platform
11.7.22. Next steps

11.8. UNINSTALLING A CLUSTER ON VSPHERE THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE
11.8.1. Removing a cluster that uses installer-provisioned infrastructure

CHAPTER 12. INSTALLING ON VMC
12.1. INSTALLING A CLUSTER ON VMC
12.1.1. Setting up VMC for vSphere
12.1.1.1. VMC Sizer tool
12.1.2. vSphere prerequisites
12.1.3. Internet access for OpenShift Container Platform
12.1.4. VMware vSphere infrastructure requirements
12.1.5. vCenter requirements
   Required vCenter account privileges
   Using OpenShift Container Platform with vMotion
   Cluster resources
   Cluster limits
   Networking requirements
   Required IP Addresses
   DNS records
12.1.6. Generating an SSH private key and adding it to the agent
12.1.7. Obtaining the installation program
12.1.8. Adding vCenter root CA certificates to your system trust
12.1.9. Deploying the cluster
12.1.10. Installing the OpenShift CLI by downloading the binary
   12.1.10.1. Installing the OpenShift CLI on Linux
   12.1.10.2. Installing the OpenShift CLI on Windows
   12.1.10.3. Installing the OpenShift CLI on macOS
12.1.11. Logging in to the cluster by using the CLI
12.1.12. Creating registry storage
   12.1.12.1. Image registry removed during installation
   12.1.12.2. Image registry storage configuration
      12.1.12.2.1. Configuring registry storage for VMware vSphere
      12.1.12.2.2. Configuring block registry storage for VMware vSphere
   12.1.13. Backing up VMware vSphere volumes
12.1.14. Telemetry access for OpenShift Container Platform
12.1.15. Next steps

12.2. INSTALLING A CLUSTER ON VMC WITH CUSTOMIZATIONS
12.2.1. Setting up VMC for vSphere
12.2.1.1. VMC Sizer tool
12.2.2. vSphere prerequisites
12.2.3. Internet access for OpenShift Container Platform
12.2.4. VMware vSphere infrastructure requirements
12.2.5. vCenter requirements
   Required vCenter account privileges
   Using OpenShift Container Platform with vMotion
   Cluster resources
   Cluster limits
   Networking requirements
   Required IP Addresses
12.2.6. Generating an SSH private key and adding it to the agent
12.2.7. Obtaining the installation program
12.2.8. Adding vCenter root CA certificates to your system trust
12.2.9. Creating the installation configuration file
  12.2.9.1. Installation configuration parameters
    12.2.9.1.1. Required configuration parameters
    12.2.9.1.2. Network configuration parameters
    12.2.9.1.3. Optional configuration parameters
    12.2.9.1.4. Additional VMware vSphere configuration parameters
    12.2.9.1.5. Optional VMware vSphere machine pool configuration parameters
  12.2.9.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster
  12.2.9.3. Configuring the cluster-wide proxy during installation
12.2.10. Deploying the cluster
12.2.11. Installing the OpenShift CLI by downloading the binary
  12.2.11.1. Installing the OpenShift CLI on Linux
  12.2.11.2. Installing the OpenShift CLI on Windows
  12.2.11.3. Installing the OpenShift CLI on macOS
12.2.12. Logging in to the cluster by using the CLI
12.2.13. Creating registry storage
  12.2.13.1. Image registry removed during installation
  12.2.13.2. Image registry storage configuration
    12.2.13.2.1. Configuring registry storage for VMware vSphere
    12.2.13.2.2. Configuring block registry storage for VMware vSphere
  12.2.14. Backing up VMware vSphere volumes
12.2.15. Telemetry access for OpenShift Container Platform
12.2.16. Next steps
12.3. INSTALLING A CLUSTER ON VMC WITH NETWORK CUSTOMIZATIONS
  12.3.1. Setting up VMC for vSphere
  12.3.1.1. VMC Sizer tool
  12.3.2. vSphere prerequisites
  12.3.3. Internet access for OpenShift Container Platform
  12.3.4. VMware vSphere infrastructure requirements
  12.3.5. vCenter requirements
    Required vCenter account privileges
    Using OpenShift Container Platform with vMotion
    Cluster resources
    Cluster limits
    Networking requirements
    Required IP Addresses
    DNS records
  12.3.6. Generating an SSH private key and adding it to the agent
  12.3.7. Obtaining the installation program
  12.3.8. Adding vCenter root CA certificates to your system trust
  12.3.9. Creating the installation configuration file
    12.3.9.1. Installation configuration parameters
      12.3.9.1.1. Required configuration parameters
      12.3.9.1.2. Network configuration parameters
      12.3.9.1.3. Optional configuration parameters
      12.3.9.1.4. Additional VMware vSphere configuration parameters
      12.3.9.1.5. Optional VMware vSphere machine pool configuration parameters
    12.3.9.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster
    12.3.9.3. Configuring the cluster-wide proxy during installation
12.3.10. Network configuration phases 1976
12.3.11. Specifying advanced network configuration 1977
defaultNetwork object configuration 1979
Configuration for the OpenShift SDN CNI cluster network provider 1979
Configuration for the OVN-Kubernetes CNI cluster network provider 1980
kubeProxyConfig object configuration 1981
12.3.13. Deploying the cluster 1982
12.3.14. Installing the OpenShift CLI by downloading the binary 1984
12.3.14.1. Installing the OpenShift CLI on Linux 1984
12.3.14.2. Installing the OpenShift CLI on Windows 1985
12.3.14.3. Installing the OpenShift CLI on macOS 1985
12.3.15. Logging in to the cluster by using the CLI 1986
12.3.16. Creating registry storage 1986
12.3.16.1. Image registry removed during installation 1986
12.3.16.2. Image registry storage configuration 1987
12.3.16.2.1. Configuring registry storage for VMware vSphere 1987
12.3.16.2.2. Configuring block registry storage for VMware vSphere 1988
12.3.17. Backing up VMware vSphere volumes 1989
12.3.18. Telemetry access for OpenShift Container Platform 1990
12.3.19. Next steps 1990

12.4. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK 1990
12.4.1. Setting up VMC for vSphere 1990
12.4.1.1. VMC Sizer tool 1992
12.4.2. vSphere prerequisites 1993
12.4.3. About installations in restricted networks 1993
12.4.3.1. Additional limits 1993
12.4.4. Internet access for OpenShift Container Platform 1993
12.4.5. VMware vSphere infrastructure requirements 1994
12.4.6. vCenter requirements 1995
  Required vCenter account privileges 1995
  Using OpenShift Container Platform with vMotion 1998
  Cluster resources 1999
  Cluster limits 1999
  Networking requirements 1999
  Required IP Addresses 1999
  DNS records 2000
12.4.7. Generating an SSH private key and adding it to the agent 2000
12.4.8. Adding vCenter root CA certificates to your system trust 2002
12.4.9. Creating the RHCOS image for restricted network installations 2002
12.4.10. Creating the installation configuration file 2003
  12.4.10.1. Installation configuration parameters 2005
  12.4.10.1.1. Required configuration parameters 2006
  12.4.10.1.2. Network configuration parameters 2007
  12.4.10.1.3. Optional configuration parameters 2009
  12.4.10.1.4. Additional VMware vSphere configuration parameters 2014
  12.4.10.1.5. Optional VMware vSphere machine pool configuration parameters 2015
12.4.10.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster 2016
12.4.10.3. Configuring the cluster-wide proxy during installation 2018
12.4.11. Deploying the cluster 2019
12.4.12. Installing the OpenShift CLI by downloading the binary 2021
  12.4.12.1. Installing the OpenShift CLI on Linux 2021
12.4.12.2. Installing the OpenShift CLI on Windows
12.4.12.3. Installing the OpenShift CLI on macOS
12.4.13. Logging in to the cluster by using the CLI
12.4.14. Disabling the default OperatorHub sources
12.4.15. Creating registry storage
  12.4.15.1. Image registry removed during installation
  12.4.15.2. Image registry storage configuration
    12.4.15.2.1. Configuring registry storage for VMware vSphere
12.4.16. Telemetry access for OpenShift Container Platform
12.4.17. Next steps

12.5. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE
12.5.1. Setting up VMC for vSphere
  12.5.1.1. VMC Sizer tool
  12.5.2. vSphere prerequisites
12.5.3. Internet access for OpenShift Container Platform
12.5.4. VMware vSphere infrastructure requirements
12.5.5. Machine requirements for a cluster with user-provisioned infrastructure
  12.5.5.1. Required machines
  12.5.5.2. Network connectivity requirements
  12.5.5.3. Minimum resource requirements
  12.5.5.4. Certificate signing requests management
12.5.6. Creating the user-provisioned infrastructure
  12.5.6.1. Networking requirements for user-provisioned infrastructure
    Network topology requirements
    Load balancers
    NTP configuration
  12.5.6.2. User-provisioned DNS requirements
12.5.7. Generating an SSH private key and adding it to the agent
12.5.8. Obtaining the installation program
12.5.9. Manually creating the installation configuration file
  12.5.9.1. Sample install-config.yaml file for VMware vSphere
  12.5.9.2. Configuring the cluster-wide proxy during installation
12.5.10. Creating the Kubernetes manifest and Ignition config files
12.5.11. Extracting the infrastructure name
12.5.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.5.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.5.14. Disk partitioning
  Creating a separate /var partition
12.5.15. Installing the OpenShift CLI by downloading the binary
  12.5.15.1. Installing the OpenShift CLI on Linux
  12.5.15.2. Installing the OpenShift CLI on Windows
  12.5.15.3. Installing the OpenShift CLI on macOS
12.5.16. Creating the cluster
12.5.17. Logging in to the cluster by using the CLI
12.5.18. Approving the certificate signing requests for your machines
12.5.19. Initial Operator configuration
  12.5.19.1. Image registry removed during installation
  12.5.19.2. Image registry storage configuration
    12.5.19.2.1. Configuring registry storage for VMware vSphere
  12.5.19.2.2. Configuring storage for the image registry in non-production clusters
  12.5.19.2.3. Configuring block registry storage for VMware vSphere
12.5.20. Completing installation on user-provisioned infrastructure
12.5.21. Backing up VMware vSphere volumes
12.6. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE AND NETWORK CUSTOMIZATIONS

12.6.1. Setting up VMC for vSphere
  12.6.1.1. VMC Sizer tool
  12.6.2. vSphere prerequisites
12.6.3. Internet access for OpenShift Container Platform
12.6.4. VMware vSphere infrastructure requirements
12.6.5. Machine requirements for a cluster with user-provisioned infrastructure
  12.6.5.1. Required machines
  12.6.5.2. Network connectivity requirements
  12.6.5.3. Minimum resource requirements
  12.6.5.4. Certificate signing requests management
12.6.6. Creating the user-provisioned infrastructure
  12.6.6.1. Networking requirements for user-provisioned infrastructure
    Network topology requirements
    Load balancers
    NTP configuration
  12.6.6.2. User-provisioned DNS requirements
12.6.7. Generating an SSH private key and adding it to the agent
12.6.8. Obtaining the installation program
12.6.9. Manually creating the installation configuration file
  12.6.9.1. Sample install-config.yaml file for VMware vSphere
  12.6.9.2. Configuring the cluster-wide proxy during installation
12.6.10. Specifying advanced network configuration
12.6.11. Cluster Network Operator configuration
  12.6.11.1. Cluster Network Operator configuration object
    defaultNetwork object configuration
    Configuration for the OpenShift SDN CNI cluster network provider
    Configuration for the OVN-Kubernetes CNI cluster network provider
    kubeProxyConfig object configuration
12.6.12. Creating the Ignition config files
12.6.13. Extracting the infrastructure name
12.6.14. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.6.15. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.6.16. Disk partitioning
    Creating a separate /var partition
12.6.17. Creating the cluster
12.6.18. Logging in to the cluster by using the CLI
12.6.19. Approving the certificate signing requests for your machines
12.6.20. Initial Operator configuration
  12.6.20.1. Image registry removed during installation
  12.6.20.2. Image registry storage configuration
  12.6.20.2.1. Configuring block registry storage for VMware vSphere
12.6.21. Completing installation on user-provisioned infrastructure
12.6.22. Backing up VMware vSphere volumes
12.6.23. Telemetry access for OpenShift Container Platform
12.6.24. Next steps

12.7. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE
12.7.2. vSphere prerequisites
12.7.3. About installations in restricted networks
12.7.3.1. Additional limits
12.7.4. Internet access for OpenShift Container Platform
12.7.5. VMware vSphere infrastructure requirements
12.7.6. Machine requirements for a cluster with user-provisioned infrastructure
12.7.6.1. Required machines
12.7.6.2. Network connectivity requirements
12.7.6.3. Minimum resource requirements
12.7.6.4. Certificate signing requests management
12.7.7. Creating the user-provisioned infrastructure
12.7.7.1. Networking requirements for user-provisioned infrastructure
   Network topology requirements
   Load balancers
   NTP configuration
12.7.7.2. User-provisioned DNS requirements
12.7.8. Generating an SSH private key and adding it to the agent
12.7.9. Manually creating the installation configuration file
12.7.9.1. Sample install-config.yaml file for VMware vSphere
12.7.9.2. Configuring the cluster-wide proxy during installation
12.7.10. Creating the Kubernetes manifest and Ignition config files
12.7.11. Extracting the infrastructure name
12.7.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.7.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere
12.7.14. Disk partitioning
   Creating a separate /var partition
12.7.15. Creating the cluster
12.7.16. Logging in to the cluster by using the CLI
12.7.17. Approving the certificate signing requests for your machines
12.7.18. Initial Operator configuration
12.7.18.1. Disabling the default OperatorHub sources
12.7.18.2. Image registry storage configuration
   12.7.18.2.1. Configuring registry storage for VMware vSphere
   12.7.18.2.2. Configuring storage for the image registry in non-production clusters
   12.7.18.2.3. Configuring block registry storage for VMware vSphere
12.7.19. Completing installation on user-provisioned infrastructure
12.7.20. Backing up VMware vSphere volumes
12.7.21. Telemetry access for OpenShift Container Platform
12.7.22. Next steps
12.7.8. UNINSTALLING A CLUSTER ON VMC
12.8. UNINSTALLING A CLUSTER ON VMC
12.8.1. Removing a cluster that uses installer-provisioned infrastructure

CHAPTER 13. INSTALLING ON ANY PLATFORM

13.1. INSTALLING A CLUSTER ON ANY PLATFORM
13.1.1. Prerequisites
13.1.2. Internet access for OpenShift Container Platform
13.1.3. Machine requirements for a cluster with user-provisioned infrastructure
   13.1.3.1. Required machines
   13.1.3.2. Network connectivity requirements
   13.1.3.3. Minimum resource requirements
   13.1.3.4. Certificate signing requests management
13.1.4. Creating the user-provisioned infrastructure
   13.1.4.1. Networking requirements for user-provisioned infrastructure
Network topology requirements
Load balancers
NTP configuration
13.1.4.2. User-provisioned DNS requirements
13.1.5. Generating an SSH private key and adding it to the agent
13.1.6. Obtaining the installation program
13.1.7. Installing the OpenShift CLI by downloading the binary
   13.1.7.1. Installing the OpenShift CLI on Linux
   13.1.7.2. Installing the OpenShift CLI on Windows
   13.1.7.3. Installing the OpenShift CLI on macOS
13.1.8. Manually creating the installation configuration file
   13.1.8.1. Configuring the cluster-wide proxy during installation
13.1.9. Configuring a three-node cluster
13.1.10. Creating the Kubernetes manifest and Ignition config files
13.1.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process
   13.1.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image
   13.1.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting
   13.1.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration
      13.1.11.3.1. Using advanced networking options for PXE and ISO installations
      13.1.11.3.2. Disk partitioning
         13.1.11.3.2.1. Creating a separate /var partition
         13.1.11.3.2.2. Retaining existing partitions
      13.1.11.3.3. Identifying Ignition configs
         13.1.11.3.3.1. Embedding an Ignition config in the RHCOS ISO
      13.1.11.3.4. Advanced RHCOS installation reference
         Routing and bonding options at RHCOS boot prompt
         coreos.inst boot options for ISO or PXE install
         coreos-installer options for ISO install
13.1.12. Creating the cluster
13.1.13. Logging in to the cluster by using the CLI
13.1.14. Approving the certificate signing requests for your machines
13.1.15. Initial Operator configuration
   13.1.15.1. Disabling the default OperatorHub sources
   13.1.15.2. Image registry removed during installation
   13.1.15.3. Image registry storage configuration
      13.1.15.3.1. Configuring registry storage for bare metal and other manual installations
      13.1.15.3.2. Configuring storage for the image registry in non-production clusters
   13.1.15.3.3. Configuring block registry storage
13.1.16. Completing installation on user-provisioned infrastructure
13.1.17. Telemetry access for OpenShift Container Platform
13.1.18. Next steps

CHAPTER 14. INSTALLATION CONFIGURATION

14.1. SUPPORTED INSTALLATION METHODS FOR DIFFERENT PLATFORMS
14.2. CUSTOMIZING NODES
   14.2.1. Adding day-1 kernel arguments
   14.2.2. Adding kernel modules to nodes
      14.2.2.1. Building and testing the kernel module container
      14.2.2.2. Provisioning a kernel module to OpenShift Container Platform
      14.2.2.2.1. Provision kernel modules via a MachineConfig object
   14.2.3. Encrypting disks during installation
      14.2.3.1. Enabling TPM v2 disk encryption
      14.2.3.2. Enabling Tang disk encryption
14.2.4. Configuring a RAID-enabled data volume
14.2.5. Configuring chrony time service
14.2.6. Additional resources
14.3. AVAILABLE CLUSTER CUSTOMIZATIONS
14.3.1. Cluster configuration resources
14.3.2. Operator configuration resources
14.3.3. Additional configuration resources
14.3.4. Informational Resources
14.3.5. Updating the global cluster pull secret
14.4. CONFIGURING YOUR FIREWALL
14.4.1. Configuring your firewall for OpenShift Container Platform
14.5. CONFIGURING A PRIVATE CLUSTER
14.5.1. About private clusters
   DNS
   Ingress Controller
   API server
14.5.2. Setting DNS to private
14.5.3. Setting the Ingress Controller to private
14.5.4. Restricting the API server to private
14.5.5. Setting the Ingress Controller to private
14.5.6. Restricting the API server to private

CHAPTER 15. VALIDATING AN INSTALLATION ........................................ 2229
15.1. REVIEWING THE INSTALLATION LOG .......................... 2229
15.2. VIEWING THE IMAGE PULL SOURCE .......................... 2229
15.3. GETTING CLUSTER VERSION, STATUS, AND UPDATE DETAILS 2230
15.4. QUERYING THE STATUS OF THE CLUSTER NODES BY USING THE CLI 2232
15.5. REVIEWING THE CLUSTER STATUS FROM THE OPENSOURCE CONTAINER PLATFORM WEB CONSOLE 2233
15.6. REVIEWING THE CLUSTER STATUS FROM RED HAT OPENSOURCE CONTAINER PLATFORM MANAGER 2233
15.7. CHECKING CLUSTER RESOURCE AVAILABILITY AND UTILIZATION 2234
15.8. LISTING ALERTS THAT ARE FIRING .......................... 2235
15.9. NEXT STEPS .................................................. 2236

CHAPTER 16. TROUBLESHOOTING INSTALLATION ISSUES ...................... 2237
16.1. PREREQUISITES ............................................. 2237
16.2. GATHERING LOGS FROM A FAILED INSTALLATION ............. 2237
16.3. MANUALLY GATHERING LOGS WITH SSH ACCESS TO YOUR HOST(S) 2238
16.4. MANUALLY GATHERING LOGS WITHOUT SSH ACCESS TO YOUR HOST(S) 2239
16.5. GETTING DEBUG INFORMATION FROM THE INSTALLATION PROGRAM 2239
16.6. REINSTALLING THE OPENSOURCE CONTAINER PLATFORM CLUSTER 2240

CHAPTER 17. SUPPORT FOR FIPS CRYPTOGRAPHY ............................... 2241
17.1. FIPS VALIDATION IN OPENSOURCE CONTAINER PLATFORM .......................... 2241
17.2. FIPS SUPPORT IN COMPONENTS THAT THE CLUSTER USES
   17.2.1. etcd .................................................. 2242
   17.2.2. Storage ............................................. 2242
   17.2.3. Runtimes ............................................. 2242
17.3. INSTALLING A CLUSTER IN FIPS MODE .......................... 2242
CHAPTER 1. MIRRroring IMages For A DISCONNECTED INSTALLATION

You can use the procedures in this section to ensure your clusters only use container images that satisfy your organizational controls on external content. Before you install a cluster on infrastructure that you provision in a restricted network, you must mirror the required container images into that environment. To mirror container images, you must have a registry for mirroring.

**IMPORTANT**

You must have access to the internet to obtain the necessary container images. In this procedure, you place your mirror registry on a mirror host that has access to both your network and the Internet. If you do not have access to a mirror host, use the Mirroring an Operator catalog procedure to copy images to a device you can move across network boundaries with.

1.1. PREREQUISITES

- You must have a container image registry that supports Docker v2-2 in the location that will host the OpenShift Container Platform cluster, such as one of the following registries:
  - Red Hat Quay
  - JFrog Artifactory
  - Sonatype Nexus Repository
  - Harbor

If you have an entitlement to Red Hat Quay, see the documentation on deploying Red Hat Quay for proof-of-concept purposes or by using the Quay Operator. If you need additional assistance selecting and installing a registry, contact your sales representative or Red Hat support.

- If you do not already have an existing solution for a container image registry, subscribers of OpenShift Container Platform are provided a mirror registry for Red Hat OpenShift. The mirror registry for Red Hat OpenShift is included with your subscription and is a small-scale container registry that can be used to mirror the required container images of OpenShift Container Platform in disconnected installations.

1.2. ABOUT THE MIRROR REGISTRY

You can mirror the images that are required for OpenShift Container Platform installation and subsequent product updates to a container mirror registry such as Red Hat Quay, JFrog Artifactory, Sonatype Nexus Repository, or Harbor. If you do not have access to a large-scale container registry, you can use the mirror registry for Red Hat OpenShift, a small-scale container registry included with OpenShift Container Platform subscriptions.

You can use any container registry that supports Docker v2-2, such as Red Hat Quay, the mirror registry for Red Hat OpenShift, Artifactory, Sonatype Nexus Repository, or Harbor. Regardless of your chosen registry, the procedure to mirror content from Red Hat hosted sites on the internet to an isolated image registry is the same. After you mirror the content, you configure each cluster to retrieve this content from your mirror registry.
IMPORTANT

The internal registry of the OpenShift Container Platform cluster cannot be used as the target registry because it does not support pushing without a tag, which is required during the mirroring process.

If choosing a container registry that is not the mirror registry for Red Hat OpenShift, it must be reachable by every machine in the clusters that you provision. If the registry is unreachable, installation, updating, or normal operations such as workload relocation might fail. For that reason, you must run mirror registries in a highly available way, and the mirror registries must at least match the production availability of your OpenShift Container Platform clusters.

When you populate your mirror registry with OpenShift Container Platform images, you can follow two scenarios. If you have a host that can access both the internet and your mirror registry, but not your cluster nodes, you can directly mirror the content from that machine. This process is referred to as connected mirroring. If you have no such host, you must mirror the images to a file system and then bring that host or removable media into your restricted environment. This process is referred to as disconnected mirroring.

For mirrored registries, to view the source of pulled images, you must review the Trying to access log entry in the CRI-O logs. Other methods to view the image pull source, such as using the crictl images command on a node, show the non-mirrored image name, even though the image is pulled from the mirrored location.

NOTE

Red Hat does not test third party registries with OpenShift Container Platform.

Additional information

For information on viewing the CRI-O logs to view the image source, see Viewing the image pull source.

1.3. PREPARING YOUR MIRROR HOST

Before you perform the mirror procedure, you must prepare the host to retrieve content and push it to the remote location.

1.3.1. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

1.3.1.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xzvf <file>

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

1.3.1.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   C:▷ path

After you install the OpenShift CLI, it is available using the oc command:

C:▷ oc <command>

1.3.1.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ``

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

1.4. CONFIGURING CREDENTIALS THAT ALLOW IMAGES TO BE MIRRORED

Create a container image registry credentials file that allows mirroring images from Red Hat to your mirror.

WARNING

Do not use this image registry credentials file as the pull secret when you install a cluster. If you provide this file when you install cluster, all of the machines in the cluster will have write access to your mirror registry.

WARNING

This process requires that you have write access to a container image registry on the mirror registry and adds the credentials to a registry pull secret.

Prerequisites

- You configured a mirror registry to use in your restricted network.
- You identified an image repository location on your mirror registry to mirror images into.
- You provisioned a mirror registry account that allows images to be uploaded to that image repository.

Procedure

Complete the following steps on the installation host:
1. Download your `registry.redhat.io` pull secret from the Red Hat OpenShift Cluster Manager and save it to a `.json` file.

2. Generate the base64-encoded user name and password or token for your mirror registry:

   $ echo -n '<user_name>:<password>' | base64
   BGVtbYk3ZHAtqXs=

   For `<user_name>` and `<password>`, specify the user name and password that you configured for your registry.

3. Make a copy of your pull secret in JSON format:

   $ cat ./pull-secret.text | jq . > <path>/<pull_secret_file_in_json>

   Specify the path to the folder to store the pull secret in and a name for the JSON file that you create.

   The contents of the file resemble the following example:

   ```json
   {
     "auths": {
       "cloud.openshift.com": {
         "auth": "b3BlbnNo...",
         "email": "you@example.com"
       },
       "quay.io": {
         "auth": "b3BlbnNo...",
         "email": "you@example.com"
       },
       "registry.connect.redhat.com": {
         "auth": "NTE3Njg5Nj...",
         "email": "you@example.com"
       },
       "registry.redhat.io": {
         "auth": "NTE3Njg5Nj...",
         "email": "you@example.com"
       }
     }
   }
   
   For `<mirror_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, `registry.example.com` or `registry.example.com:5000`.
For `<credentials>`, specify the base64-encoded user name and password for the mirror registry.

The file resembles the following example:

```json
{
  "auths": {
    "registry.example.com": {
      "auth": "BGVtbYk3ZHAtqXs=",
      "email": "you@example.com"
    },
    "cloud.openshift.com": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "quay.io": {
      "auth": "b3BlbnNo...",
      "email": "you@example.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    },
    "registry.redhat.io": {
      "auth": "NTE3Njg5Nj...",
      "email": "you@example.com"
    }
  }
}
```

1.5. MIRROR REGISTRY FOR RED HAT OPENSSHIFT

The `mirror registry for Red Hat OpenShift` is a small and streamlined container registry that you can use as a target for mirroring the required container images of OpenShift Container Platform for disconnected installations.

If you already have a container image registry, such as Red Hat Quay, you can skip these steps and go straight to `Mirroring the OpenShift Container Platform image repository`.

Prerequisites

- An OpenShift Container Platform subscription.
- Red Hat Enterprise Linux (RHEL) 8 with Podman 3.3 and OpenSSL installed.
- Fully qualified domain name for the Red Hat Quay service, which must resolve through a DNS server.
- Passwordless `sudo` access on the target host.
- Key-based SSH connectivity on the target host. SSH keys are automatically generated for local installs. For remote hosts, you must generate your own SSH keys.
- 2 or more vCPUs.
8 GB of RAM.

About 6.8 GB for OpenShift Container Platform 4.6 Release images, or about 696 GB for OpenShift Container Platform 4.6 Release images and OpenShift Container Platform 4.6 Red Hat Operator images. Up to 1 TB per stream or more is suggested.

IMPORTANT

These requirements are based on local testing results with only Release images and Operator images tested. Storage requirements can vary based on your organization’s needs. Some users might require more space, for example, when they mirror multiple z-streams. You can use standard Red Hat Quay functionality to remove unnecessary images and free up space.

1.5.1. Mirror registry for Red Hat OpenShift introduction

For disconnected deployments of OpenShift Container Platform, a container registry is required to carry out the installation of the clusters. To run a production-grade registry service on such a cluster, you must create a separate registry deployment to install the first cluster. The mirror registry for Red Hat OpenShift addresses this need and is included in every OpenShift subscription. It is available for download on the OpenShift console Downloads page.

The mirror registry for Red Hat OpenShift allows users to install a small-scale version of Red Hat Quay and its required components using the mirror-registry command line interface (CLI) tool. The mirror registry for Red Hat OpenShift is deployed automatically with pre-configured local storage and a local database. It also includes auto-generated user credentials and access permissions with a single set of inputs and no additional configuration choices to get started.

The mirror registry for Red Hat OpenShift provides a pre-determined network configuration and reports deployed component credentials and access URLs upon success. A limited set of optional configuration inputs like fully qualified domain name (FQDN) services, superuser name and password, and custom TLS certificates are also provided. This provides users with a container registry so that they can easily create an offline mirror of all OpenShift Container Platform release content when running OpenShift Container Platform in restricted network environments.

The mirror registry for Red Hat OpenShift is limited to hosting images that are required to install a disconnected OpenShift Container Platform cluster, such as Release images or Red Hat Operator images. It uses local storage on your Red Hat Enterprise Linux (RHEL) machine, and storage supported by RHEL is supported by the mirror registry for Red Hat OpenShift. Content built by customers should not be hosted by the mirror registry for Red Hat OpenShift.

Unlike Red Hat Quay, the mirror registry for Red Hat OpenShift is not a highly-available registry and only local file system storage is supported. Using the mirror registry for Red Hat OpenShift with more than one cluster is discouraged, because multiple clusters can create a single point of failure when updating your cluster fleet. It is advised to leverage the mirror registry for Red Hat OpenShift to install a cluster that can host a production-grade, highly-available registry such as Red Hat Quay, which can serve OpenShift Container Platform content to other clusters.

Use of the mirror registry for Red Hat OpenShift is optional if another container registry is already available in the install environment.

1.5.2. Mirroring on a local host with mirror registry for Red Hat OpenShift
This procedure explains how to install the mirror registry for Red Hat OpenShift on a local host using the mirror-registry installer tool. By doing so, users can create a local host registry running on port 443 for the purpose of storing a mirror of OpenShift Container Platform images.

**NOTE**

Installing the mirror registry for Red Hat OpenShift using the mirror-registry CLI tool makes several changes to your machine. After installation, a /etc/quay-install directory is created, which has installation files, local storage, and the configuration bundle. Trusted SSH keys are generated in case the deployment target is the local host, and systemd files on the host machine are set up to ensure that container runtimes are persistent. Additionally, an initial user named init is created with an automatically generated password. All access credentials are printed at the end of the install routine.

**Procedure**

1. Download the mirror-registry.tar.gz package for the latest version of the mirror registry for Red Hat OpenShift found on the OpenShift console Downloads page.

2. Install the mirror registry for Red Hat OpenShift on your local host with your current user account by using the mirror-registry tool. For a full list of available flags, see "mirror registry for Red Hat OpenShift flags".

   ```
   $ sudo ./mirror-registry install \ 
   --quayHostname <host_example_com> \ 
   --quayRoot <example_directory_name>
   ```

3. Use the user name and password generated during installation to log into the registry by running the following command:

   ```
   $ podman login --authfile pull-secret.txt \ 
   -u init \ 
   -p <password> \ 
   <host_example_com>:8443 \ 
   --tls-verify=false
   ```

   You can avoid running --tls-verify=false by configuring your system to trust the generated rootCA certificates. See "Using SSL to protect connections to Red Hat Quay" and "Configuring the system to trust the certificate authority" for more information.

**NOTE**

You can also log in by accessing the UI at https://<host.example.com>:8443 after installation.

4. You can mirror OpenShift Container Platform images after logging in. Depending on your needs, see either the "Mirroring the OpenShift Container Platform image repository" or the "Mirroring an Operator catalog" sections of this document.
1.5.3. Mirroring on a remote host with mirror registry for Red Hat OpenShift

This procedure explains how to install the mirror registry for Red Hat OpenShift on a remote host using the mirror-registry tool. By doing so, users can create a registry to hold a mirror of OpenShift Container Platform images.

NOTE

Installing the mirror registry for Red Hat OpenShift using the mirror-registry CLI tool makes several changes to your machine. After installation, a /etc/quay-install directory is created, which has installation files, local storage, and the configuration bundle. Trusted SSH keys are generated in case the deployment target is the local host, and systemd files on the host machine are set up to ensure that container runtimes are persistent. Additionally, an initial user named init is created with an automatically generated password. All access credentials are printed at the end of the install routine.

Procedure

1. Download the mirror-registry.tar.gz package for the latest version of the mirror registry for Red Hat OpenShift found on the OpenShift console Downloads page.

2. Install the mirror registry for Red Hat OpenShift on your local host with your current user account by using the mirror-registry tool. For a full list of available flags, see "mirror registry for Red Hat OpenShift flags".

   $ sudo ./mirror-registry install -v \ 
   --targetHostname <host_example_com> \ 
   --targetUsername <example_user> \ 
   -k ~/.ssh/my_ssh_key \ 
   --quayHostname <host_example_com> \ 
   --quayRoot <example_directory_name>

3. Use the user name and password generated during installation to log into the mirror registry by running the following command:

   $ podman login --authfile pull-secret.txt \ 
   -u init \ 
   -p <password> \ 
   <host_example_com>:8443 \ 
   --tls-verify=false

   You can avoid running --tls-verify=false by configuring your system to trust the generated rootCA certificates. See "Using SSL to protect connections to Red Hat Quay" and "Configuring the system to trust the certificate authority" for more information.
You can also log in by accessing the UI at https://<host.example.com>:8443 after installation.

4. You can mirror OpenShift Container Platform images after logging in. Depending on your needs, see either the "Mirroring the OpenShift Container Platform image repository" or the "Mirroring an Operator catalog" sections of this document.

If there are issues with images stored by the mirror registry for Red Hat OpenShift due to storage layer problems, you can remirror the OpenShift Container Platform images, or reinstall mirror registry on more stable storage.

1.6. UPGRADING THE MIRROR REGISTRY FOR RED HAT OPENSHIFT

You can upgrade the mirror registry for Red Hat OpenShift from your local host by running the following command:

$ sudo ./mirror-registry upgrade

Users who upgrade the mirror registry for Red Hat OpenShift with the ./mirror-registry upgrade flag must include the same credentials used when creating their mirror registry. For example, if you installed the mirror registry for Red Hat OpenShift with --quayHostname <host_example_com> and --quayRoot <example_directory_name>, you must include that string to properly upgrade the mirror registry.

1.6.1. Uninstalling the mirror registry for Red Hat OpenShift

You can uninstall the mirror registry for Red Hat OpenShift from your local host by running the following command:

$ sudo ./mirror-registry uninstall -v \ 
--quayRoot <example_directory_name>

Deleting the mirror registry for Red Hat OpenShift will prompt the user before deletion. You can use --autoApprove to skip this prompt.

Users who install the mirror registry for Red Hat OpenShift with the --quayRoot flag must include the --quayRoot flag when uninstalling. For example, if you installed the mirror registry for Red Hat OpenShift with --quayRoot example_directory_name, you must include that string to properly uninstall the mirror registry.

1.6.2. Mirror registry for Red Hat OpenShift flags

The following flags are available for the mirror registry for Red Hat OpenShift:
### Flags

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>--autoApprove</code></td>
<td>A boolean value that disables interactive prompts. If set to <code>true</code>, the <code>quayRoot</code> directory is automatically deleted when uninstalling the mirror registry. Defaults to <code>false</code> if left unspecified.</td>
</tr>
<tr>
<td><code>--initPassword</code></td>
<td>The password of the init user created during Quay installation. Must be at least eight characters and contain no whitespace.</td>
</tr>
<tr>
<td><code>--initUser string</code></td>
<td>Shows the username of the initial user. Defaults to <code>init</code> if left unspecified.</td>
</tr>
<tr>
<td><code>--quayHostname</code></td>
<td>The fully-qualified domain name of the mirror registry that clients will use to contact the registry. Equivalent to <code>SERVER_HOSTNAME</code> in the Quay <code>config.yaml</code>. Must resolve by DNS. Defaults to <code>&lt;targetHostname&gt;:8443</code> if left unspecified.</td>
</tr>
<tr>
<td><code>--quayRoot, -r</code></td>
<td>The directory where container image layer and configuration data is saved, including <code>rootCA.key</code>, <code>rootCA.pem</code>, and <code>rootCA.srl</code> certificates. Requires about 6.8 GB for OpenShift Container Platform 4.6 Release images, or about 696 GB for OpenShift Container Platform 4.6 Release images and OpenShift Container Platform 4.6 Red Hat Operator images. Defaults to <code>/etc/quay-install</code> if left unspecified.</td>
</tr>
<tr>
<td><code>--ssh-key, -k</code></td>
<td>The path of your SSH identity key. Defaults to <code>~/.ssh/quay_installer</code> if left unspecified.</td>
</tr>
<tr>
<td><code>--sslCert</code></td>
<td>The path to the SSL/TLS public key/certificate. Defaults to <code>{quayRoot}/quay-config</code> and is auto-generated if left unspecified.</td>
</tr>
<tr>
<td><code>--sslCheckSkip</code></td>
<td>Skips the check for the certificate hostname against the <code>SERVER_HOSTNAME</code> in the <code>config.yaml</code> file.</td>
</tr>
<tr>
<td><code>--sslKey</code></td>
<td>The path to the SSL/TLS private key used for HTTPS communication. Defaults to <code>{quayRoot}/quay-config</code> and is auto-generated if left unspecified.</td>
</tr>
<tr>
<td><code>--targetHostname, -H</code></td>
<td>The hostname of the target you want to install Quay to. Defaults to <code>$HOST</code>, for example, a local host, if left unspecified.</td>
</tr>
<tr>
<td><code>--targetUsername, -u</code></td>
<td>The user on the target host which will be used for SSH. Defaults to <code>$USER</code>, for example, the current user if left unspecified.</td>
</tr>
<tr>
<td><code>--verbose, -v</code></td>
<td>Shows debug logs and Ansible playbook outputs.</td>
</tr>
<tr>
<td><code>--version</code></td>
<td>Shows the version for the mirror registry for Red Hat OpenShift</td>
</tr>
</tbody>
</table>

1. `--quayHostname` must be modified if the public DNS name of your system is different from the local hostname.
2. `--sslCheckSkip` is used in cases when the mirror registry is set behind a proxy and the exposed hostname is different from the internal Quay hostname. It can also be used when users do not want the certificates to be validated against the provided Quay hostname during installation.

Additional resources

- Using SSL to protect connections to Red Hat Quay
- Configuring the system to trust the certificate authority
- Mirroring the OpenShift Container Platform image repository
- Mirroring an Operator catalog

1.7. MIRRORING THE OPENSSHIFT CONTAINER PLATFORM IMAGE REPOSITORY

Mirror the OpenShift Container Platform image repository to your registry to use during cluster installation or upgrade.

Prerequisites

- Your mirror host has access to the Internet.
- You configured a mirror registry to use in your restricted network and can access the certificate and credentials that you configured.
- You downloaded the pull secret from the Red Hat OpenShift Cluster Manager and modified it to include authentication to your mirror repository.
- If you use self-signed certificates that do not set a Subject Alternative Name, you must precede the `oc` commands in this procedure with `GODEBUG=x509ignoreCN=0`. If you do not set this variable, the `oc` commands will fail with the following error:

```
x509: certificate relies on legacy Common Name field, use SANs or temporarily enable
Common Name matching with GODEBUG=x509ignoreCN=0
```

Procedure

Complete the following steps on the mirror host:

1. Review the OpenShift Container Platform downloads page to determine the version of OpenShift Container Platform that you want to install and determine the corresponding tag on the Repository Tags page.

2. Set the required environment variables:
   a. Export the release version:

   ```
   $ OCP_RELEASE=<release_version>
   ```

   For `<release_version>`, specify the tag that corresponds to the version of OpenShift Container Platform to install, such as 4.5.4.

   b. Export the local registry name and host port:

   ```
For `<local_registry_host_name>`, specify the registry domain name for your mirror repository, and for `<local_registry_host_port>`, specify the port that it serves content on.

c. Export the local repository name:

```
$ LOCAL_REPOSITORY='<local_repository_name>'
```

For `<local_repository_name>`, specify the name of the repository to create in your registry, such as `ocp4/openshift4`.

d. Export the name of the repository to mirror:

```
$ PRODUCT_REPO='openshift-release-dev'
```

For a production release, you must specify `openshift-release-dev`.

e. Export the path to your registry pull secret:

```
$ LOCAL_SECRET_JSON='<path_to_pull_secret>'
```

For `<path_to_pull_secret>`, specify the absolute path to and file name of the pull secret for your mirror registry that you created.

f. Export the release mirror:

```
$ RELEASE_NAME="ocp-release"
```

For a production release, you must specify `ocp-release`.

g. Export the type of architecture for your server, such as `x86_64`:

```
$ ARCHITECTURE=<server_architecture>
```

h. Export the path to the directory to host the mirrored images:

```
$ REMOVABLE_MEDIA_PATH=<path> ①
```

① Specify the full path, including the initial forward slash (/) character.

3. Mirror the version images to the mirror registry:

- If your mirror host does not have internet access, take the following actions:
  
i. Connect the removable media to a system that is connected to the internet.

   ```
   $ oc adm release mirror -a $LOCAL_SECRET_JSON \
   --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-${ARCHITECTURE} \
   --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
   ```

   Review the images and configuration manifests to mirror:

   ```
   $ oc adm release mirror -a $LOCAL_SECRET_JSON \
   --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-${ARCHITECTURE} \
   --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
   ```
iii. Record the entire `imageContentSources` section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the `imageContentSources` section to the `install-config.yaml` file during installation.

iv. Mirror the images to a directory on the removable media:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}+
${ARCHITECTURE}
```

v. Take the media to the restricted network environment and upload the images to the local container registry.

```
$ oc image mirror -a ${LOCAL_SECRET_JSON} --from-
dir=${REMOVABLE_MEDIA_PATH}/mirror
"file://openshift/release:${OCP_RELEASE}"*
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}
```

For `REMOVABLE_MEDIA_PATH`, you must use the same path that you specified when you mirrored the images.

- If the local container registry is connected to the mirror host, take the following actions:

  i. Directly push the release images to the local registry by using following command:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON}\n  --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCPRELEASE}-
  ${ARCHITECTURE}\n  --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}\n  --to-release-
  image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
  ${ARCHITECTURE}
```

This command pulls the release information as a digest, and its output includes the `imageContentSources` data that you require when you install your cluster.

ii. Record the entire `imageContentSources` section from the output of the previous command. The information about your mirrors is unique to your mirrored repository, and you must add the `imageContentSources` section to the `install-config.yaml` file during installation.

**NOTE**

The image name gets patched to Quay.io during the mirroring process, and the podman images will show Quay.io in the registry on the bootstrap virtual machine.
4. To create the installation program that is based on the content that you mirrored, extract it and pin it to the release:

- If your mirror host does not have Internet access, run the following command:
  
  ```bash
  $ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-install
  "${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}"
  
  $ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-install
  "${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCPRELEASE}"
  
  IMPORTANT
  
  To ensure that you use the correct images for the version of OpenShift Container
  Platform that you selected, you must extract the installation program from the mirrored
  content.

  You must perform this step on a machine with an active Internet connection.

  If you are in a disconnected environment, use the `--image` flag as part of must-gather and
  point to the payload image.

1.8. THE CLUSTER SAMPLES OPERATOR IN A DISCONNECTED ENVIRONMENT

In a disconnected environment, you must take additional steps after you install a cluster to configure the
Cluster Samples Operator.

1.9. NEXT STEPS

- **Mirror** the OperatorHub images for the Operators that you want to install in your cluster.

- **Install** a cluster on infrastructure that you provision in your restricted network, such as on
  VMware vSphere, bare metal, or Amazon Web Services.

1.10. ADDITIONAL RESOURCES

- See **Gathering data about specific features** for more information about using must-gather.
CHAPTER 2. INSTALLING ON AWS

2.1. CONFIGURING AN AWS ACCOUNT

Before you can install OpenShift Container Platform, you must configure an Amazon Web Services (AWS) account.

2.1.1. Configuring Route 53

To install OpenShift Container Platform, the Amazon Web Services (AWS) account you use must have a dedicated public hosted zone in your Route 53 service. This zone must be authoritative for the domain. The Route 53 service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through AWS or another source.

   NOTE

   If you purchase a new domain through AWS, it takes time for the relevant DNS changes to propagate. For more information about purchasing domains through AWS, see Registering Domain Names Using Amazon Route 53 in the AWS documentation.

2. If you are using an existing domain and registrar, migrate its DNS to AWS. See Making Amazon Route 53 the DNS Service for an Existing Domain in the AWS documentation.

3. Create a public hosted zone for your domain or subdomain. See Creating a Public Hosted Zone in the AWS documentation.
   Use an appropriate root domain, such as openshiftcorp.com, or subdomain, such as clusters.openshiftcorp.com.

4. Extract the new authoritative name servers from the hosted zone records. See Getting the Name Servers for a Public Hosted Zone in the AWS documentation.

5. Update the registrar records for the AWS Route 53 name servers that your domain uses. For example, if you registered your domain to a Route 53 service in a different accounts, see the following topic in the AWS documentation: Adding or Changing Name Servers or Glue Records.

6. If you are using a subdomain, add its delegation records to the parent domain. This gives Amazon Route 53 responsibility for the subdomain. Follow the delegation procedure outlined by the DNS provider of the parent domain. See Creating a subdomain that uses Amazon Route 53 as the DNS service without migrating the parent domain in the AWS documentation for an example high level procedure.

2.1.1.1. Ingress Operator endpoint configuration for AWS Route 53

If you install in either Amazon Web Services (AWS) GovCloud (US) US-West or US-East region, the Ingress Operator uses us-gov-west-1 region for Route53 and tagging API clients.

The Ingress Operator uses https://tagging.us-gov-west-1.amazonaws.com as the tagging API endpoint if a tagging custom endpoint is configured that includes the string ‘us-gov-east-1’.
For more information on AWS GovCloud (US) endpoints, see the Service Endpoints in the AWS documentation about GovCloud (US).

**IMPORTANT**

Private, disconnected installations are not supported for AWS GovCloud when you install in the **us-gov-east-1** region.

**Example Route 53 configuration**

platform:
  aws:
    region: us-gov-west-1
    serviceEndpoints:
      - name: ec2
        url: https://ec2.us-gov-west-1.amazonaws.com
      - name: elasticloadbalancing
        url: https://elasticloadbalancing.us-gov-west-1.amazonaws.com
      - name: route53
        url: https://route53.us-gov.amazonaws.com
      - name: tagging
        url: https://tagging.us-gov-west-1.amazonaws.com

1. Route 53 defaults to **https://route53.us-gov.amazonaws.com** for both AWS GovCloud (US) regions.

2. Only the US-West region has endpoints for tagging. Omit this parameter if your cluster is in another region.

**2.1.2. AWS account limits**

The OpenShift Container Platform cluster uses a number of Amazon Web Services (AWS) components, and the default Service Limits affect your ability to install OpenShift Container Platform clusters. If you use certain cluster configurations, deploy your cluster in certain AWS regions, or run multiple clusters from your account, you might need to request additional resources for your AWS account.

The following table summarizes the AWS components whose limits can impact your ability to install and run OpenShift Container Platform clusters.
<table>
<thead>
<tr>
<th>Component</th>
<th>Number of clusters available by default</th>
<th>Default AWS limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Limits</td>
<td>Varies</td>
<td>Varies</td>
<td>By default, each cluster creates the following instances:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- One bootstrap machine, which is removed after installation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Three control plane nodes (also known as the master nodes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Three worker nodes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>These instance type counts are within a new account's default limit. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, review your account limits to ensure that your cluster can deploy the machines that you need.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>In most regions, the bootstrap and worker machines uses an m4.large machines and the control plane machines use m4.xlarge instances. In some regions, including all regions that do not support these instance types, m5.large and m5.xlarge instances are used instead.</td>
</tr>
<tr>
<td>Elastic IPs (EIPs)</td>
<td>0 to 1</td>
<td>5 EIPs per account</td>
<td>To provision the cluster in a highly available configuration, the installation program creates a public and private subnet for each availability zone within a region. Each private subnet requires a NAT Gateway, and each NAT gateway requires a separate elastic IP. Review the AWS region map to determine how many availability zones are in each region. To take advantage of the default high availability, install the cluster in a region with at least three availability zones. To install a cluster in a region with more than five availability zones, you must increase the EIP limit.</td>
</tr>
<tr>
<td>Virtual Private Clouds (VPCs)</td>
<td>5</td>
<td>5 VPCs per region</td>
<td>Each cluster creates its own VPC.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

To use the us-east-1 region, you must increase the EIP limit for your account.
### Chapter 2. Installing on AWS

#### 2.1.3. Required AWS permissions

**NOTE**

Your IAM user must have the permission `tag:GetResources` in the region `us-east-1` to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.
When you attach the **AdministratorAccess** policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

**Example 2.1. Required EC2 permissions for installation**

- `tag:TagResources`
- `tag:UntagResources`
- `ec2:AllocateAddress`
- `ec2:AssociateAddress`
- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:CopyImage`
- `ec2:CreateNetworkInterface`
- `ec2:AttachNetworkInterface`
- `ec2:CreateSecurityGroup`
- `ec2:CreateTags`
- `ec2:CreateVolume`
- `ec2:DeleteSecurityGroup`
- `ec2:DeleteSnapshot`
- `ec2:DeleteTags`
- `ec2:DeregisterImage`
- `ec2:DescribeAccountAttributes`
- `ec2:DescribeAddresses`
- `ec2:DescribeAvailabilityZones`
- `ec2:DescribeDhcpOptions`
- `ec2:DescribeImages`
- `ec2:DescribeInstanceAttribute`
- `ec2:DescribeInstanceCreditSpecifications`
- `ec2:DescribeInstances`
- `ec2:DescribeInternetGateways`
- `ec2:DescribeKeyPairs`
Example 2.2. Required permissions for creating network resources during installation

- ec2:AssociateDhcpOptions
- ec2:AssociateRouteTable
- ec2:AttachInternetGateway
- ec2:CreateDhcpOptions
- ec2:CreateInternetGateway
- `ec2:CreateNatGateway`
- `ec2:CreateRoute`
- `ec2:CreateRouteTable`
- `ec2:CreateSubnet`
- `ec2:CreateVpc`
- `ec2:CreateVpcEndpoint`
- `ec2:ModifySubnetAttribute`
- `ec2:ModifyVpcAttribute`

**NOTE**

If you use an existing VPC, your account does not require these permissions for creating network resources.

---

**Example 2.3. Required Elastic Load Balancing permissions (ELB) for installation**

- `elasticloadbalancing:AddTags`
- `elasticloadbalancing:ApplySecurityGroupsToLoadBalancer`
- `elasticloadbalancing:AttachLoadBalancerToSubnets`
- `elasticloadbalancing:ConfigureHealthCheck`
- `elasticloadbalancing:CreateLoadBalancer`
- `elasticloadbalancing:CreateLoadBalancerListeners`
- `elasticloadbalancing:DeleteLoadBalancer`
- `elasticloadbalancing:DeregisterInstancesFromLoadBalancer`
- `elasticloadbalancing:DescribeInstanceHealth`
- `elasticloadbalancing:DescribeLoadBalancerAttributes`
- `elasticloadbalancing:DescribeLoadBalancers`
- `elasticloadbalancing:DescribeTags`
- `elasticloadbalancing:ModifyLoadBalancerAttributes`
- `elasticloadbalancing:RegisterInstancesWithLoadBalancer`
- `elasticloadbalancing:SetLoadBalancerPoliciesOfListener`

---

**Example 2.4. Required Elastic Load Balancing permissions (ELBv2) for installation**
• elasticloadbalancing:AddTags
• elasticloadbalancing:CreateListener
• elasticloadbalancing:CreateLoadBalancer
• elasticloadbalancing:CreateTargetGroup
• elasticloadbalancing:DeleteLoadBalancer
• elasticloadbalancing:DeregisterTargets
• elasticloadbalancing:DescribeListeners
• elasticloadbalancing:DescribeLoadBalancerAttributes
• elasticloadbalancing:DescribeLoadBalancers
• elasticloadbalancing:DescribeTargetGroupAttributes
• elasticloadbalancing:DescribeTargetHealth
• elasticloadbalancing:ModifyLoadBalancerAttributes
• elasticloadbalancing:ModifyTargetGroup
• elasticloadbalancing:ModifyTargetGroupAttributes
• elasticloadbalancing:RegisterTargets

Example 2.5. Required IAM permissions for installation

• iam:AddRoleToInstanceProfile
• iam:CreateInstanceProfile
• iam:CreateRole
• iam:DeleteInstanceProfile
• iam:DeleteRole
• iam:DeleteRolePolicy
• iam:GetInstanceProfile
• iam:GetRole
• iam:GetRolePolicy
• iam:GetUser
• iam:ListInstanceProfilesForRole
• iam:ListRoles
• iam:ListUsers
- iam:PassRole
- iam:PutRolePolicy
- iam:RemoveRoleFromInstanceProfile
- iam:SimulatePrincipalPolicy
- iam:TagRole

NOTE

If you have not created an elastic load balancer (ELB) in your AWS account, the IAM user also requires the `iam:CreateServiceLinkedRole` permission.

Example 2.6. Required Route 53 permissions for installation

- route53:ChangeResourceRecordSets
- route53:ChangeTagsForResource
- route53:CreateHostedZone
- route53:DeleteHostedZone
- route53:GetChange
- route53:GetHostedZone
- route53:GetHostZoneComment
- route53:GetHostedZones
- route53:GetHostedZonesByName
- route53:GetResourceRecordSets
- route53:GetTagsForResource
- route53:ListTagsForResource
- route53:UpdateHostedZoneComment

Example 2.7. Required S3 permissions for installation

- s3:CreateBucket
- s3:DeleteBucket
- s3:GetAccelerateConfiguration
- s3:GetBucketAcl
- s3:GetBucketCors
- s3:GetBucketLocation
- s3:GetBucketLogging
Example 2.8. S3 permissions that cluster Operators require

- s3:DeleteObject
- s3:GetObject
- s3:GetObjectAcl
- s3:GetObjectTagging
- s3:GetObjectVersion
- s3:PutObject
- s3:PutObjectAcl
- s3:PutObjectTagging

Example 2.9. Required permissions to delete base cluster resources

- autoscaling:DescribeAutoScalingGroups
- ec2:DeleteNetworkInterface
- ec2:DeleteVolume
- elasticloadbalancing:DeleteTargetGroup
- elasticloadbalancing:DescribeTargetGroups
Example 2.10. Required permissions to delete network resources

- ec2:DeleteDhcpOptions
- ec2:DeleteInternetGateway
- ec2:DeleteNatGateway
- ec2:DeleteRoute
- ec2:DeleteRouteTable
- ec2:DeleteSubnet
- ec2:DeleteVpc
- ec2:DeleteVpcEndpoints
- ec2:DetachInternetGateway
- ec2:DisassociateRouteTable
- ec2:ReplaceRouteTableAssociation

NOTE

If you use an existing VPC, your account does not require these permissions to delete network resources.

Example 2.11. Additional IAM and S3 permissions that are required to create manifests

- iam:DeleteAccessKey
- iam:DeleteUser
- iam:DeleteUserPolicy
• `iam:GetUserPolicy`
• `iam:GetUserPolicy`
• `iam:ListAccessKeys`
• `iam:ListAccessKeys`
• `iam:PutUserPolicy`
• `iam:PutUserPolicy`
• `iam:TagUser`
• `iam:TagUser`
• `iam:GetUserPolicy`
• `iam:GetUserPolicy`
• `s3:PutBucketPublicAccessBlock`
• `s3:PutBucketPublicAccessBlock`
• `s3:PutLifecycleConfiguration`
• `s3:HeadBucket`
• `s3:ListBucketMultipartUploads`
• `s3:AbortMultipartUpload`

NOTE
If you are managing your cloud provider credentials with mint mode, the IAM user also requires the `iam:CreateAccessKey` and `iam:CreateUser` permissions.

Example 2.12. Optional permission for quota checks for installation
• `servicequotas:ListAWSDefaultServiceQuotas`

2.1.4. Creating an IAM user

Each Amazon Web Services (AWS) account contains a root user account that is based on the email address you used to create the account. This is a highly-privileged account, and it is recommended to use it for only initial account and billing configuration, creating an initial set of users, and securing the account.

Before you install OpenShift Container Platform, create a secondary IAM administrative user. As you complete the Creating an IAM User in Your AWS Account procedure in the AWS documentation, set the following options:

Procedure

1. Specify the IAM user name and select **Programmatic access**.

2. Attach the **AdministratorAccess** policy to ensure that the account has sufficient permission to create the cluster. This policy provides the cluster with the ability to grant credentials to each OpenShift Container Platform component. The cluster grants the components only the credentials that they require.
NOTE

While it is possible to create a policy that grants the all of the required AWS permissions and attach it to the user, this is not the preferred option. The cluster will not have the ability to grant additional credentials to individual components, so the same credentials are used by all components.

3. Optional: Add metadata to the user by attaching tags.

4. Confirm that the user name that you specified is granted the AdministratorAccess policy.

5. Record the access key ID and secret access key values. You must use these values when you configure your local machine to run the installation program.

IMPORTANT

You cannot use a temporary session token that you generated while using a multi-factor authentication device to authenticate to AWS when you deploy a cluster. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials.

Additional resources

- See Manually creating IAM for AWS for steps to set the Cloud Credential Operator (CCO) to manual mode prior to installation. Use this mode in environments where the cloud identity and access management (IAM) APIs are not reachable, or if you prefer not to store an administrator-level credential secret in the cluster kube-system project.

2.1.5. Supported AWS regions

You can deploy an OpenShift Container Platform cluster to the following public regions:

NOTE

Your IAM user must have the permission tag:GetResources in the region us-east-1 to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

- af-south-1 (Cape Town)
- ap-east-1 (Hong Kong)
- ap-northeast-1 (Tokyo)
- ap-northeast-2 (Seoul)
- ap-northeast-3 (Osaka)
- ap-south-1 (Mumbai)
- ap-southeast-1 (Singapore)
- ap-southeast-2 (Sydney)
• ca-central-1 (Central)
• eu-central-1 (Frankfurt)
• eu-north-1 (Stockholm)
• eu-south-1 (Milan)
• eu-west-1 (Ireland)
• eu-west-2 (London)
• eu-west-3 (Paris)
• me-south-1 (Bahrain)
• sa-east-1 (São Paulo)
• us-east-1 (N. Virginia)
• us-east-2 (Ohio)
• us-west-1 (N. California)
• us-west-2 (Oregon)

The following AWS GovCloud regions are supported:
• us-gov-west-1
• us-gov-east-1

2.1.6. Next steps
• Install an OpenShift Container Platform cluster:
  • Quickly install a cluster with default options on installer-provisioned infrastructure
  • Install a cluster with cloud customizations on installer-provisioned infrastructure
  • Install a cluster with network customizations on installer-provisioned infrastructure
  • Installing a cluster on user-provisioned infrastructure in AWS by using CloudFormation templates

2.2. MANUALLY CREATING IAM FOR AWS

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster kube-system namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

2.2.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your
organization by setting different values for the `credentialsMode` parameter in the `install-config.yaml` file.

If you prefer not to store an administrator-level credential secret in the cluster `kube-system` project, you can choose one of the following options when installing OpenShift Container Platform:

- **Manage cloud credentials manually:**
  You can set the `credentialsMode` parameter for the CCO to `Manual` to manage cloud credentials manually. Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

- **Remove the administrator-level credential secret after installing OpenShift Container Platform with mint mode:**
  If you are using the CCO with the `credentialsMode` parameter set to `Mint`, you can remove or rotate the administrator-level credential after installing OpenShift Container Platform. Mint mode is the default configuration for the CCO. This option requires the presence of the administrator-level credential during an installation. The administrator-level credential is used during the installation to mint other credentials with some permissions granted. The original credential secret is not stored in the cluster permanently.

  **NOTE**
  Prior to a non z-stream upgrade, you must reinstate the credential secret with the administrator-level credential. If the credential is not present, the upgrade might be blocked.

**Additional resources**

- To learn how to rotate or remove the administrator-level credential secret after installing OpenShift Container Platform, see [Rotating or removing cloud provider credentials](#).

- For a detailed description of all available CCO credential modes and their supported platforms, see [Cloud Credential Operator](#).

### 2.2.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster `kube-system` namespace.

**Procedure**

1. Change to the directory that contains the installation program and create the `install-config.yaml` file:

   ```
   $ openshift-install create install-config --dir <installation_directory>
   ```

2. Edit the `install-config.yaml` configuration file so that it contains the `credentialsMode` parameter set to `Manual`. 
Example install-config.yaml configuration file

```yaml
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual
compute:
  - architecture: amd64
    hyperthreading: Enabled
...
```

1. This line is added to set the credentialsMode parameter to Manual.

3. To generate the manifests, run the following command from the directory that contains the installation program:

```
$ openshift-install create manifests --dir <installation_directory>
```

1. For <installation_directory>, specify the directory name to store the files that the installation program creates.

4. Remove the admin credential secret created using your local cloud credentials. This removal prevents your admin credential from being stored in the cluster:

```
$ rm mycluster/openshift/99_cloud-creds-secret.yaml
```

5. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your openshift-install binary is built to use:

```
$ openshift-install version
```

Example output

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

6. Locate all CredentialsRequest objects in this release image that target the cloud you are deploying on:

```
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --credentials-requests --cloud=aws
```

This displays the details for each request.

Sample CredentialsRequest object

```yaml
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: cloud-credential-operator-iam-ro
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
```
Create YAML files for secrets in the `openshift-install` manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the `spec.secretRef` for each `credentialsRequest`. The format for the secret data varies for each cloud provider.

From the directory that contains the installation program, proceed with your cluster creation:

```
$ openshift-install create cluster --dir <installation_directory>
```

### IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state. For details, see the Upgrading clusters with manually maintained credentials section of the installation content for your cloud provider.

### 2.2.3. Admin credentials root secret format

Each cloud provider uses a credentials root secret in the `kube-system` namespace by convention, which is then used to satisfy all credentials requests and create their respective secrets. This is done either by minting new credentials, with `mint mode`, or by copying the credentials root secret, with `passthrough mode`.

The format for the secret varies by cloud, and is also used for each `CredentialsRequest` secret.

**Amazon Web Services (AWS) secret format**

```
apiVersion: v1
description: Secret
metadata:
  namespace: kube-system
  name: aws-creds
stringData:
  aws_access_key_id: <AccessKeyId>
  aws_secret_access_key: <SecretAccessKey>
```

### 2.2.4. Upgrading clusters with manually maintained credentials

If credentials are added in a future release, the Cloud Credential Operator (CCO) `upgradable` status for a cluster with manually maintained credentials changes to `false`. For minor release, for example, from 4.5 to 4.6, this status prevents you from upgrading until you have addressed any updated permissions.
For z-stream releases, for example, from 4.5.10 to 4.5.11, the upgrade is not blocked, but the credentials must still be updated for the new release.

Use the Administrator perspective of the web console to determine if the CCO is upgradeable.

1. Navigate to Administration → Cluster Settings.
2. To view the CCO status details, click cloud-credential in the Cluster Operators list.
3. If the Upgradeable status in the Conditions section is False, examine the credentialsRequests for the new release and update the manually maintained credentials on your cluster to match before upgrading.

In addition to creating new credentials for the release image that you are upgrading to, you must review the required permissions for existing credentials and accommodate any new permissions requirements for existing components in the new release. The CCO cannot detect these mismatches and will not set upgradable to false in this case.

The Manually creating IAM section of the installation content for your cloud provider explains how to obtain and use the credentials required for your cloud.

### 2.2.5. Mint mode

Mint mode is the default and recommended Cloud Credential Operator (CCO) credentials mode for OpenShift Container Platform. In this mode, the CCO uses the provided administrator-level cloud credential to run the cluster. Mint mode is supported for AWS, GCP, and Azure.

In mint mode, the admin credential is stored in the kube-system namespace and then used by the CCO to process the CredentialsRequest objects in the cluster and create users for each with specific permissions.

The benefits of mint mode include:

- Each cluster component has only the permissions it requires
- Automatic, on-going reconciliation for cloud credentials, including additional credentials or permissions that might be required for upgrades

One drawback is that mint mode requires admin credential storage in a cluster kube-system secret.

### 2.2.6. Mint Mode with removal or rotation of the admin credential

Currently, this mode is only supported on AWS.

In this mode, a user installs OpenShift Container Platform with an admin credential just like the normal mint mode. However, this mode removes the admin credential secret from the cluster post-installation.

The administrator can have the Cloud Credential Operator make its own request for a read-only credential that allows it to verify if all CredentialsRequest objects have their required permissions, thus the admin credential is not required unless something needs to be changed. After the associated credential is removed, it can be destroyed on the underlying cloud, if desired.

Prior to upgrade, the admin credential should be restored. In the future, upgrade might be blocked if the credential is not present.

The admin credential is not stored in the cluster permanently.
This mode still requires the **admin** credential in the cluster for brief periods of time. It also requires manually re-instating the secret with **admin** credentials for each upgrade.

### 2.2.7. Next steps

- Install an OpenShift Container Platform cluster:
  - Installing a cluster quickly on AWS with default options on installer-provisioned infrastructure
  - Install a cluster with cloud customizations on installer-provisioned infrastructure
  - Install a cluster with network customizations on installer-provisioned infrastructure
  - Installing a cluster on user-provisioned infrastructure in AWS by using CloudFormation templates

### 2.3. INSTALLING A CLUSTER QUICKLY ON AWS

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) that uses the default configuration options.

#### 2.3.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an AWS account to host the cluster.

**IMPORTANT**

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you must **configure it to allow the sites** that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can **manually create and maintain IAM credentials**. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

#### 2.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 2.3.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name> 
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.
2. Start the `ssh-agent` process as a background task:

   
   ```
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
   
   NOTE
   
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.
   ```

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>
   ```

   Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

2.3.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.
The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 2.3.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> --log-level=info
   ```

   **1** For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`. 

     Chapter 2. Installing on AWS
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Provide values at the prompts:

a. Optional: Select an SSH key to use to access your cluster machines.

b. Select **aws** as the platform to target.

c. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

b. **NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

d. Select the AWS region to deploy the cluster to.

e. Select the base domain for the Route 53 service that you configured for your cluster.

f. Enter a descriptive name for your cluster.

g. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

b. **NOTE**

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

d. Select the AWS region to deploy the cluster to.

e. Select the base domain for the Route 53 service that you configured for your cluster.

f. Enter a descriptive name for your cluster.

g. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

b. **NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the **kubeadmin** user, display in your terminal.

**Example output**

...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"  
INFO Time elapsed: 36m22s

NOTE
The cluster access and credential information also outputs to <installation_directory>/.openshift_install.log when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the AdministratorAccess policy from the IAM account that you used to install the cluster.

NOTE
The elevated permissions provided by the AdministratorAccess policy are required only during installation.

Additional resources
- See Configuration and credential file settings in the AWS documentation for more information about AWS profile and credential configuration.

2.3.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

2.3.6.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   
   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:
   
   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

2.3.6.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
   
   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
2.3.6.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

2.3.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami
2.3.8. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

   ```bash
   $ cat <installation_directory>/auth/kubeadmin-password
   
   NOTE
   Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/openshift_install.log` log file on the installation host.
   
2. List the OpenShift Container Platform web console route:

   ```bash
   $ oc get routes -n openshift-console | grep 'console-openshift'
   
   NOTE
   Alternatively, you can obtain the OpenShift Container Platform route from the `<installation_directory>/openshift_install.log` log file on the installation host.
   
   Example output

   |
   console     console-openshift-console.apps.<cluster_name>.<base_domain>            console
   https   reencrypt/Redirect   None

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the `kubeadmin` user.

**Additional resources**

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

2.3.9. Telemetry access for OpenShift Container Platform
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

2.3.10. Next steps

- Validating an installation.
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If necessary, you can remove cloud provider credentials.

2.4. INSTALLING A CLUSTER ON AWS WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a customized cluster on infrastructure that the installation program provisions on Amazon Web Services (AWS). To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

2.4.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an AWS account to host the cluster.

IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see Managing Access Keys for IAM Users in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

2.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.
You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access **Quay.io** to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 2.4.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your **ssh-agent** and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>  
```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.
NOTE
If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```bash
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```bash
$ ssh-add <path>/<file_name>
```

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 2.4.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 2.4.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select AWS as the platform to target.

iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

iv. Select the AWS region to deploy the cluster to.

v. Select the base domain for the Route 53 service that you configured for your cluster.

vi. Enter a descriptive name for your cluster.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the install-config.yaml file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the install-config.yaml file so that you can use it to install multiple clusters.

IMPORTANT

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

2.4.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.

NOTE

After installation, you cannot modify these parameters in the install-config.yaml file.
IMPORTANT

The openshift-install command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

2.4.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Table 2.1. Required parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td>apiVersion</td>
</tr>
<tr>
<td>baseDomain</td>
</tr>
<tr>
<td>metadata</td>
</tr>
<tr>
<td>metadata.name</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td><strong>platform</strong></td>
</tr>
</tbody>
</table>
| **pullSecret** | Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io. | {  
  "auths": {  
    "cloud.openshift.com": {  
      "auth": "b3Blb=",
      "email": "you@example.com"
    },
    "quay.io": {  
      "auth": "b3Blb=",
      "email": "you@example.com"
    }
  }
} |

### 2.4.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>networking</strong></td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the **networking** object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.networkType</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An IPv4 network.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
2.4.5.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 2.3. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following “Machine-pool” table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <strong>compute.platform</strong> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
### fips

Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
</tbody>
</table>

### imageContentSources

Sources and repositories for the release-image content.

<table>
<thead>
<tr>
<th>imageContentSources</th>
<th>Sources and repositories for the release-image content.</th>
<th>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>

### publish

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

<table>
<thead>
<tr>
<th>publish</th>
<th>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</th>
<th>Internal or External. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</th>
</tr>
</thead>
</table>
The SSH key or keys to authenticate access your cluster machines.

**NOTE**
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 2.4.5.1.4. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

**Table 2.4. Optional AWS parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.aws.amiID</code></td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.iops</code></td>
<td>The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.</td>
<td>Integer, for example 4000.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.size</code></td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example 500.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.type</code></td>
<td>The type of the root volume.</td>
<td>Valid AWS EBS volume type, such as <code>io1</code>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.type</code></td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid AWS instance type, such as <code>c5.9xlarge</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>compute.platform.aws.zones</td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <code>us-east-1c</code>, in a YAML sequence.</td>
</tr>
<tr>
<td>compute.aws.region</td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid AWS region, such as <code>us-east-1</code>.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.amiID</td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.type</td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid AWS instance type, such as <code>c5.9xlarge</code>.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.zones</td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <code>us-east-1c</code>, in a YAML sequence.</td>
</tr>
<tr>
<td>controlPlane.aws.region</td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid AWS region, such as <code>us-east-1</code>.</td>
</tr>
<tr>
<td>platform.aws.amiID</td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>platform.aws.serviceEndpoints.name</td>
<td>The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.</td>
<td>Valid AWS service endpoint name.</td>
</tr>
</tbody>
</table>

CHAPTER 2. INSTALLING ON AWS
### Parameter Description Values

<table>
<thead>
<tr>
<th><strong>platform.aws.serviceEndpoints.url</strong></th>
<th>The AWS service endpoint URL. The URL must use the <code>https</code> protocol and the host must trust the certificate.</th>
<th>Valid AWS service endpoint URL.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform.aws.userTags</strong></td>
<td>A map of keys and values that the installation program adds as tags to all resources that it creates.</td>
<td>Any valid YAML map, such as key-value pairs in the <code>&lt;key&gt;: &lt;value&gt;</code> format. For more information about AWS tags, see <a href="https://aws.amazon.com/documentation/ec2/tagging/">Tagging Your Amazon EC2 Resources</a> in the AWS documentation.</td>
</tr>
<tr>
<td><strong>platform.aws.subnets</strong></td>
<td>If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same <code>machineNetwork[].cidr</code> ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.</td>
<td>Valid subnet IDs.</td>
</tr>
</tbody>
</table>

### 2.4.5.2. Sample customized `install-config.yaml` file for AWS

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
credentialsMode: Mint
controlPlane: 
  hyperthreading: Enabled
  name: master
platform:
  aws:
    zones:
    - us-west-2a
    - us-west-2b
    rootVolume:
      iops: 4000
      size: 500
```
type: io1
- type: m5.xlarge
  replicas: 3
  compute:
    - hyperthreading: Enabled
      name: worker
      platform:
        aws:
          rootVolume:
            iops: 2000
            size: 500
          type: io1
          type: c5.4xlarge
          zones:
          - us-west-2c
          replicas: 3
          metadata:
            name: test-cluster
          networking:
            clusterNetwork:
            - cidr: 10.128.0.0/14
              hostPrefix: 23
            machineNetwork:
            - cidr: 10.0.0.0/16
              networkType: OpenShiftSDN
              serviceNetwork:
              - 172.30.0.0/16
              platform:
                aws:
                  region: us-west-2
                  userTags:
                    adminContact: jdoe
                    costCenter: 7536
                    amiID: ami-96c6f8f7
                    serviceEndpoints:
                    - name: ec2
                      url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
                    fips: false
                    sshKey: ssh-ed25519 AAAA...
                    pullSecret: '{"auths": ...}'

1. 10. 11. 16. Required. The installation program prompts you for this value.

2. Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

3. 7. If you do not provide these parameters and values, the installation program provides the default value.

4. The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift
Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as **m4.2xlarge** or **m5.2xlarge**, for your machines if you disable simultaneous multithreading.

To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to **2000**.

The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

### 2.4.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

**Prerequisites**

- You have an existing **install-config.yaml** file.
You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

If your cluster is on AWS, you added the `ec2.<region>.amazonaws.com`, `elasticloadbalancing.<region>.amazonaws.com`, and `s3.<region>.amazonaws.com` endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
   noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCS trust bundle. The
The installation program does not support the proxy **readinessEndpoints** field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

### 2.4.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

#### Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

#### Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \1
   --log-level=info 2
   ```

   1. For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
   2. To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the **kubeadmin** user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the **AdministratorAccess** policy from the IAM account that you used to install the cluster.
2.4.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

2.4.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   
   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

2.4.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
   
   ```shell
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

   ```shell
   C:\> oc <command>
   ```

### 2.4.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```shell
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

   ```shell
   $ oc <command>
   ```

### 2.4.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

$ oc whoami

**Example output**

system:admin

### 2.4.9. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

   $ cat <installation_directory>/auth/kubeadmin-password

   **NOTE**

   Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/.openshift_install.log` log file on the installation host.

2. List the OpenShift Container Platform web console route:

   $ oc get routes -n openshift-console | grep 'console-openshift'

   **NOTE**

   Alternatively, you can obtain the OpenShift Container Platform route from the `<installation_directory>/.openshift_install.log` log file on the installation host.

   **Example output**

   console   console-openshift-console.apps.<cluster_name>.<base_domain>   console
   https   reencrypt/Redirect   None
3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the `kubeadmin` user.

Additional resources
- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

### 2.4.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources
- See [About remote health monitoring](#) for more information about the Telemetry service.

### 2.4.11. Next steps
- Validating an installation.
- Customize your cluster.
- If necessary, you can [opt out of remote health reporting](#).
- If necessary, you can [remove cloud provider credentials](#).

### 2.5. INSTALLING A CLUSTER ON AWS WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only `kubeProxy` configuration parameters in a running cluster.

#### 2.5.1. Prerequisites
- Review details about the [OpenShift Container Platform installation and update](#) processes.
- Configure an AWS account to host the cluster.
If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you must [configure it to allow the sites](#) that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can [manually create and maintain IAM credentials](#). Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

### 2.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 2.5.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.
NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   $ ssh-keygen -t ed25519 -N " \n   -f <path>/<file_name> 1

   1 Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   $ eval "$(ssh-agent -s)"

   Example output

   Agent pid 31874

   NOTE
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   $ ssh-add <path>/<file_name> 1

   Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

2.5.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

2.5.5. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

Phase 1
After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`  
  For more information on these fields, refer to "Installation configuration parameters".

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

Phase 2

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

2.5.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.
   
   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>  
   ```

   1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:
   i. Optional: Select an SSH key to use to access your cluster machines.

   **NOTE**

   For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

   ii. Select AWS as the platform to target.

   iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

   iv. Select the AWS region to deploy the cluster to.

   v. Select the base domain for the Route 53 service that you configured for your cluster.

   vi. Enter a descriptive name for your cluster.

   vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the **Installation configuration parameters** section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 2.5.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.
IMPORTANT

The *openshift-install* command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

2.5.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 2.5. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <em>install-config.yaml</em> content. The current version is <strong>v1</strong>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <em>baseDomain</em> and <em>metadata.name</em> parameter values that uses the <em>&lt;metadata.name&gt;</em>. <em>&lt;baseDomain&gt;</em> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere.** For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform</strong></td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>pullSecret</strong></td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td></td>
</tr>
</tbody>
</table>

```json
{
    "auths":{
        "cloud.openshift.com":{
            "auth":"b3Blb=",
            "email":"you@example.com"
        },
        "quay.io":{
            "auth":"b3Blb=",
            "email":"you@example.com"
        }
    }
}
```

### 2.5.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 2.6. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>networking</strong></td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusternetwork</td>
<td>The IP address blocks for pods. The default value is <strong>10.128.0.0/14</strong> with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <strong>networking.clusterNetwork</strong>. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between <strong>0</strong> and <strong>32</strong>.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <strong>hostPrefix</strong> is set to <strong>23</strong> then each node is assigned a /23 subnet out of the given <strong>cidr</strong>. A <strong>hostPrefix</strong> value of 23 provides 510 (2^{(32 - 23)} - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>The default value is <strong>23</strong>.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <strong>172.30.0.0/16</strong>. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>
**networking.machineNetwork.cidr**

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

### 2.5.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 2.7. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>compute.hyperthreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td><strong>compute.name</strong></td>
<td>Required if you use compute. The name of the machine pool.</td>
<td><strong>worker</strong></td>
</tr>
<tr>
<td><strong>compute.platform</strong></td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td><strong>aws</strong>, <strong>azure</strong>, <strong>gcp</strong>, <strong>openstack</strong>, <strong>ovirt</strong>, <strong>vsphere</strong>, or {}</td>
</tr>
<tr>
<td><strong>compute.replicas</strong></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><strong>controlPlane</strong></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><strong>controlPlane.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td><strong>fips</strong></td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td><strong>imageContentSources</strong></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><strong>imageContentSources.source</strong></td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td><strong>imageContentSources.mirrors</strong></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td><strong>publish</strong></td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 2.5.6.14. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

#### Table 2.8. Optional AWS parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.aws.amiID</code></td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.iops</code></td>
<td>The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.</td>
<td>Integer, for example <strong>4000</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.size</code></td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example <strong>500</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.type</code></td>
<td>The type of the root volume.</td>
<td>Valid <strong>AWS EBS volume type</strong>, such as <strong>io1</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.type</code></td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid <strong>AWS instance type</strong>, such as <strong>c5.9xlarge</strong>.</td>
</tr>
</tbody>
</table>
## Parameter Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>compute.platform.aws.zones</strong></td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><strong>compute.aws.region</strong></td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.amiID</strong></td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.type</strong></td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid AWS instance type, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.zones</strong></td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><strong>controlPlane.aws.region</strong></td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><strong>platform.aws.amiID</strong></td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><strong>platform.aws.serviceEndpoints.name</strong></td>
<td>The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.</td>
<td>Valid AWS service endpoint name.</td>
</tr>
</tbody>
</table>
2.5.6.2. Sample customized install-config.yaml file for AWS

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your install-config.yaml file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane:  3 4
hyperthreading: Enabled 5
name: master
platform:
  aws:
    zones:
    - us-west-2a
    - us-west-2b
  rootVolume:
    iops: 4000
    size: 500
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.aws.serviceEndpoints.url</td>
<td>The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.</td>
<td>Valid AWS service endpoint URL.</td>
</tr>
<tr>
<td>platform.aws.userTags</td>
<td>A map of keys and values that the installation program adds as tags to all resources that it creates.</td>
<td>Any valid YAML map, such as key value pairs in the &lt;key&gt;:&lt;value&gt; format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.</td>
</tr>
<tr>
<td>platform.aws.subnets</td>
<td>If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.</td>
<td>Valid subnet IDs.</td>
</tr>
</tbody>
</table>
Required. The installation program prompts you for this value.

Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift
Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

**5.8.** Whether to enable or disable simultaneous multithreading, or **hypertreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as m4.2xlarge or m5.2xlarge, for your machines if you disable simultaneous multithreading.

**6.9.** To configure faster storage for etcd, especially for larger clusters, set the storage type as **io1** and set **iops** to 2000.

**13.** The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

**14.** The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the **https** protocol and the host must trust the certificate.

**15.** Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**16.** You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

### 2.5.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

**Prerequisites**

- You have an existing **install-config.yaml** file.
You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

**NOTE**

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

If your cluster is on AWS, you added the ec2.<region>.amazonaws.com, elasticloadbalancing.<region>.amazonaws.com, and s3.<region>.amazonaws.com endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

**Procedure**

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  
     noProxy: example.com  
   additionalTrustBundle: |  
     -----BEGIN CERTIFICATE-----  
     <MY_TRUSTED_CA_CERT>  
     -----END CERTIFICATE-----  
     ...
   ```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2. A proxy URL to use for creating HTTPS connections outside the cluster.
   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
   4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The
additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE
The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster–wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE
Only the Proxy object named cluster is supported, and no additional proxies can be created.

2.5.7. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named cluster. The CR specifies the fields for the Network API in the operator.openshift.io API group.

The CNO configuration inherits the following fields during cluster installation from the Network API in the Network.config.openshift.io API group and these fields cannot be changed:

- **clusterNetwork**: IP address pools from which pod IP addresses are allocated.
- **serviceNetwork**: IP address pool for services.
- **defaultNetwork.type**: Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the defaultNetwork object in the CNO object named cluster.

2.5.7.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 2.9. Cluster Network Operator configuration object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always cluster.</td>
</tr>
</tbody>
</table>
Field | Type | Description
--- | --- | ---
**spec.clusterNetwork** | array | A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:

```
spec:
  clusterNetwork:
  - cidr: 10.128.0.0/19
    hostPrefix: 23
  - cidr: 10.128.32.0/19
    hostPrefix: 23
```

This value is ready-only and specified in the *install-config.yaml* file.

**spec.serviceNetwork** | array | A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:

```
spec:
  serviceNetwork:
  - 172.30.0.0/14
```

This value is ready-only and specified in the *install-config.yaml* file.

**spec.defaultNetwork** | object | Configures the Container Network Interface (CNI) cluster network provider for the cluster network.

**spec.kubeProxy** | object | The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.

defaultNetwork object configuration
The values for the **defaultNetwork** object are defined in the following table:

**Table 2.10. defaultNetwork object**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

Either **OpenShiftSDN** or **OVNKubernetes**. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.

**NOTE**
OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>Either OpenShiftSDN or OVNKubernetes. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>openshiftSDNConfig</th>
<th>object</th>
<th>This object is only valid for the OpenShift SDN cluster network provider.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ovnKubernetesConfig</td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

**Configuration for the OpenShift SDN CNI cluster network provider**
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

**Table 2.11. openshiftSDNConfig object**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is <strong>NetworkPolicy</strong>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The values <strong>Multitenant</strong> and <strong>Subnet</strong> are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>
The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.

This value cannot be changed after cluster installation.

The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.

If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.

On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.

Example OpenShift SDN configuration

defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789

Configuration for the OVN-Kubernetes CNI cluster network provider

The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

Table 2.12. ovnKubernetesConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultNetwork</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshiftSDNConfig</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
<td></td>
</tr>
<tr>
<td>vxlanPort</td>
<td>integer</td>
<td></td>
</tr>
</tbody>
</table>
The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.

This value cannot be changed after cluster installation.

genevePort

The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.

Example OVN-Kubernetes configuration

defaultNetwork:
type: OVNKubernetes
ovnKubernetesConfig:
  mtu: 1400
genevePort: 6081

kubeProxyConfig object configuration
The values for the `kubeProxyConfig` object are defined in the following table:

Table 2.13. `kubeProxyConfig` object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>genevePort</td>
<td>integer</td>
<td>The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>defaultNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type: OVNKubernetes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ovnKubernetesConfig:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtu: 1400</td>
<td></td>
<td></td>
</tr>
<tr>
<td>genevePort: 6081</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for <em>iptables</em> rules. The default value is <strong>30s</strong>. Valid suffixes include <strong>s</strong>, <strong>m</strong>, and <strong>h</strong> and are described in the <em>Go time</em> package documentation.</td>
</tr>
</tbody>
</table>

**NOTE**

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the `iptablesSyncPeriod` parameter is no longer necessary.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxyArguments.iptables-</td>
<td>array</td>
<td>The minimum duration before refreshing <em>iptables</em> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include <strong>s</strong>, <strong>m</strong>, and <strong>h</strong> and are described in the <em>Go time</em> package. The default value is:</td>
</tr>
</tbody>
</table>
| min-sync-period               |        | **kubeProxyConfig:**
|                               |        | proxyArguments:
|                               |        |   iptables-min-sync-period:
|                               |        |   - 0s                                                                   |

### 2.5.8. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**

- Create the `install-config.yaml` file and complete any modifications to it.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`
Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

where:

`<installation_directory>`

Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

Specify a different VXLAN port for the OpenShift SDN network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    openshiftSDNConfig:
      vxlanPort: 4800
```

4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

NOTE

For more information on using a Network Load Balancer (NLB) on AWS, see Configuring Ingress cluster traffic on AWS using a Network Load Balancer.

2.5.9. Configuring an Ingress Controller Network Load Balancer on a new AWS cluster

You can create an Ingress Controller backed by an AWS Network Load Balancer (NLB) on a new cluster.

Prerequisites

- Create the `install-config.yaml` file and complete any modifications to it.

Procedure

Create an Ingress Controller backed by an AWS NLB on a new cluster.
1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ``

   For `<installation_directory>`, specify the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a file that is named `cluster-ingress-default-ingresscontroller.yaml` in the `<installation_directory>/manifests/` directory:

   ```bash
   $ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
   ``

   For `<installation_directory>`, specify the directory name that contains the `manifests/` directory for your cluster.

   After creating the file, several network configuration files are in the `manifests/` directory, as shown:

   ```bash
   $ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
   ``

   Example output

   ```
   cluster-ingress-default-ingresscontroller.yaml
   ```

3. Open the `cluster-ingress-default-ingresscontroller.yaml` file in an editor and enter a CR that describes the Operator configuration you want:

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: IngressController
   metadata:
     creationTimestamp: null
     name: default
     namespace: openshift-ingress-operator
   spec:
     endpointPublishingStrategy:
       loadBalancer:
         scope: External
         providerParameters:
           type: AWS
           aws:
             type: NLB
   type: LoadBalancerService
   ```

4. Save the `cluster-ingress-default-ingresscontroller.yaml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-ingress-default-ingresscontroller.yaml` file. The installation program deletes the `manifests/` directory when creating the cluster.

### 2.5.10. Configuring hybrid networking with OVN–Kubernetes

You can configure your cluster to use hybrid networking with OVN–Kubernetes. This allows a hybrid

```bash
$ ./openshift-install create manifests --dir <installation_directory>
```

```bash
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

### Example output

```bash
ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

#### Cluster-ingress-default-ingresscontroller.yaml

```yaml
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
      providerParameters:
        type: AWS
        aws:
          type: NLB
    type: LoadBalancerService
```
You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.

**IMPORTANT**

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

**Prerequisites**

- You defined OVNKubernetes for the networking.networkType parameter in the install-config.yaml file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   
   where:
   
   **<installation_directory>**
   
   Specifies the name of the directory that contains the install-config.yaml file for your cluster.
   
   2. Create a stub manifest file for the advanced network configuration that is named cluster-network-03-config.yml in the <installation_directory>/manifests/ directory:

   ```
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   
   where:
   
   **<installation_directory>**
   
   Specifies the directory name that contains the manifests/ directory for your cluster.

   3. Open the cluster-network-03-config.yml file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

   **Specify a hybrid networking configuration**

   ```
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     defaultNetwork:
Specify the CIDR configuration used for nodes on the additional overlay network. The `hybridClusterNetwork` CIDR cannot overlap with the `clusterNetwork` CIDR.

Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default `4789` port. For more information on this requirement, see the Microsoft documentation on Pod-to-pod connectivity between hosts is broken.

4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

NOTE
For more information on using Linux and Windows nodes in the same cluster, see Understanding Windows container workloads.

2.5.11. Deploying the cluster
You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure
1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info
   ```

   For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```bash
INFO Install complete!
INFO To access the cluster as the system:admin user when using `oc`, run `export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig`
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the `AdministratorAccess` policy from the IAM account that you used to install the cluster.
2.5.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

2.5.12.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   
   ```
   $ tar xvzf <file>
   ```
5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

2.5.12.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
   
   ```
   C:\> path
   ```
   
   After you install the OpenShift CLI, it is available using the oc command:
   
   ```
   C:\> oc <command>
   ```

2.5.12.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   
   ```
   $ echo $PATH
   ```
   
   After you install the OpenShift CLI, it is available using the oc command:
   
   ```
   $ oc <command>
   ```

2.5.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   **Example output**

   ```bash
   system:admin
   ```

2.5.14. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

   ```bash
   $ cat <installation_directory>/auth/kubeadmin-password
   ```

   **NOTE**

   Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/openshift_install.log` log file on the installation host.

2. List the OpenShift Container Platform web console route:

   ```bash
   $ oc get routes -n openshift-console | grep 'console-openshift'
   ```

   **NOTE**

   Alternatively, you can obtain the OpenShift Container Platform route from the `<installation_directory>/openshift_install.log` log file on the installation host.

   **Example output**

   ```bash
   console console-openshift-console.apps.<cluster_name>.<base_domain> console
   https reencrypt/Redirect None
   ```
3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

### 2.5.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to **OpenShift Cluster Manager**.

After you confirm that your **OpenShift Cluster Manager** inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use **subscription watch** to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service.

### 2.5.16. Next steps

- **Validating an installation**.
- **Customize your cluster**.
- If necessary, you can **opt out of remote health reporting**.
- If necessary, you can **remove cloud provider credentials**.

## 2.6. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) in a restricted network by creating an internal mirror of the installation release content on an existing Amazon Virtual Private Cloud (VPC).

### 2.6.1. Prerequisites

- You **mirrored the images for a disconnected installation** to your registry and obtained the **imageContentSources** data for your version of OpenShift Container Platform.

  **IMPORTANT**
  
  Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You have an existing VPC in AWS. When installing to a restricted network using installer-provisioned infrastructure, you cannot use the installer-provisioned VPC. You must use a user-provisioned VPC that satisfies one of the following requirements:
  - **Contains the mirror registry**.
- Has firewall rules or a peering connection to access the mirror registry hosted elsewhere.

- You reviewed details about the OpenShift Container Platform installation and update processes.

- You configured an AWS account to host the cluster.

**IMPORTANT**

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see Managing Access Keys for IAM Users in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix) in the AWS documentation.

- If you use a firewall and plan to use the Telemetry service, you configured the firewall to allow the sites that your cluster requires access to.

**NOTE**

If you are configuring a proxy, be sure to also review this site list.

- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

### 2.6.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

#### 2.6.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The ClusterVersion status includes an Unable to retrieve available updates error.
By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

2.6.3. About using a custom VPC

In OpenShift Container Platform 4.6, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

2.6.3.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See Amazon VPC console wizard configurations and Work with VPCs and subnets in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See VPC networking components and Route tables for your VPC for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the kubernetes.io/cluster/.*: owned tag. The installation program modifies your subnets to add the kubernetes.io/cluster/.*: shared tag, so your subnets must have at least one free tag slot available for it. See Tag Restrictions in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify.
- You must enable the `enableDnsSupport` and `enableDnsHostnames` attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster’s internal DNS records. See [DNS Support in Your VPC](https://docs.aws.amazon.com/vpc/latest/userguide/vpc-dns-support.html) in the AWS documentation.

  If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the `platform.aws.hostedZone` field in the `install-config.yaml` file.

- If you use a cluster with public access, you must create a public and a private subnet for each availability zone that your cluster uses. Each availability zone can contain no more than one public and one private subnet.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2 and ELB endpoints. To resolve this, you must create a VPC endpoint and attach it to the subnet that the clusters are using. The endpoints should be named as follows:

- `ec2.<region>.amazonaws.com`
- `elasticloadbalancing.<region>.amazonaws.com`
- `s3.<region>.amazonaws.com`

### Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td><img src="https://i.imgur.com/3.png" alt="VPC" /></td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td>Public subnets</td>
<td><img src="https://i.imgur.com/3.png" alt="Public subnets" /></td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td>Internet gateway</td>
<td><img src="https://i.imgur.com/3.png" alt="Internet gateway" /></td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
</tbody>
</table>
### Network access control

- **AWS::EC2::NetworkAcl**
- **AWS::EC2::NetworkAclEntry**

You must allow the VPC to access the following ports:

<table>
<thead>
<tr>
<th>Port</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td>1024 - 65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td>0 - 65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
</tbody>
</table>

### Private subnets

- **AWS::EC2::Subnet**
- **AWS::EC2::RouteTable**
- **AWS::EC2::SubnetRouteTableAssociation**

Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

#### 2.6.3.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.
If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the `kubernetes.io/cluster/.*: shared` tag is removed from the subnets that it used.

### 2.6.3.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, Internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

### 2.6.3.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

### 2.6.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](https://github.com/openshift/openshift-4.6) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

2.6.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
```

Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:
NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   `$ ssh-add <path>/<file_name>`

   **Example output**

   `Identity added: /home/<you>/<path>/<file_name> (<computer_name>)`

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 2.6.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the `imageContentSources` values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   `$ ./openshift-install create install-config --dir <installation_directory>`

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select AWS as the platform to target.

iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

iv. Select the AWS region to deploy the cluster to.

v. Select the base domain for the Route 53 service that you configured for your cluster.

vi. Enter a descriptive name for your cluster.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Edit the install-config.yaml file to provide the additional information that is required for an installation in a restricted network.

a. Update the pullSecret value to contain the authentication information for your registry:

```yaml
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email": "you@example.com"}}}
```

For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

b. Add the additionalTrustBundle parameter and value.

```yaml
additionalTrustBundle: |
-----BEGIN CERTIFICATE-----
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
-----END CERTIFICATE-----
```
The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

c. Define the subnets for the VPC to install the cluster in:

   subnets:
   - subnet-1
   - subnet-2
   - subnet-3

d. Add the image content resources, which look like this excerpt:

   ```yaml
   imageContentSources:
     - mirrors:
       - <mirror_host_name>:<repo_name>/release
         source: quay.example.com/openshift-release-dev/ocp-release
       - mirrors:
         source: registry.example.com/ocp/release
   ```

   To complete these values, use the `imageContentSources` that you recorded during mirror registry creation.

3. Make any other modifications to the `install-config.yaml` file that you require. You can find more information about the available parameters in the Installation configuration parameters section.

4. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 2.6.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

#### NOTE

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 2.6.6.1.1. Required configuration parameters
Required installation configuration parameters are described in the following table:

### Table 2.14. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{.metadata.name}),  {.baseDomain}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
pullSecret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

2.6.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 2.15. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterN</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>etwork</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The default value is <strong>10.128.0.0/14</strong> with a host prefix of <strong>/23</strong>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given <code>cidr</code>. A <code>hostPrefix</code> value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking: serviceNetwork: - 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking: machineNetwork: - cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use <code>networking.machineNetwork</code>. An IP address block.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
</tbody>
</table>
|                                         |                                                                            | Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in. |}

2.6.6.13. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 2.16. Optional parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <strong>compute</strong>. The name of the machine pool.</td>
<td><strong>worker</strong></td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <strong>compute</strong>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <strong>controlPlane.platform</strong> parameter value.</td>
<td><strong>aws</strong>, <strong>azure</strong>, <strong>gcp</strong>, <strong>openstack</strong>, <strong>ovirt</strong>, <strong>vsphere</strong>, or <strong>{}</strong></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to <strong>2</strong>. The default value is <strong>3</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>IMPORTANT</td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>imageContentSources</th>
<th>Sources and repositories for the release-image content.</th>
<th>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><strong>Internal</strong> or <strong>External</strong>. To deploy a private cluster, which cannot be accessed from the internet, set <strong>publish</strong> to <strong>Internal</strong>. The default value is <strong>External</strong>.</td>
</tr>
</tbody>
</table>
**sshKey**

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```plaintext
sshKey:
  <key1>
  <key2>
  <key3>
```

### 2.6.6.14. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

**Table 2.17. Optional AWS parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.aws.amiID</td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.iops</td>
<td>The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.</td>
<td>Integer, for example 4000.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.size</td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example 500.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.type</td>
<td>The type of the root volume.</td>
<td>Valid AWS EBS volume type, such as io1.</td>
</tr>
<tr>
<td>compute.platform.aws.type</td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid AWS instance type, such as c5.9xlarge.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>compute.platform.aws.zones</code></td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>compute.aws.region</code></td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.amiID</code></td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.type</code></td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid AWS instance type, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.zones</code></td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>controlPlane.aws.region</code></td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><code>platform.aws.amiID</code></td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
</tbody>
</table>
## Parameter Description Values

**platform.aws.serviceEndpoints.name**  
The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.  
Valid AWS service endpoint name.

**platform.aws.serviceEndpoints.url**  
The AWS service endpoint URL. The URL must use the `https` protocol and the host must trust the certificate.  
Valid AWS service endpoint URL.

**platform.aws.userTags**  
A map of keys and values that the installation program adds as tags to all resources that it creates.  
Any valid YAML map, such as key value pairs in the `<key>: <value>` format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.

**platform.aws.subnets**  
If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same `machineNetwork[].cidr` ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.  
Valid subnet IDs.

### 2.6.6.2. Sample customized install-config.yaml file for AWS

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1  
baseDomain: example.com
```
credentialsMode: Mint
controlPlane: 
  hyperthreading: Enabled
name: master
platform:
  aws:
    zones:
      - us-west-2a
      - us-west-2b
    rootVolume:
      iops: 4000
      size: 500
      type: io1
    type: m5.xlarge
replicas: 3
compute: 
  - hyperthreading: Enabled
name: worker
platform:
  aws:
    rootVolume:
      iops: 2000
      size: 500
      type: io1
    type: c5.4xlarge
    zones:
      - us-west-2c
replicas: 3
metadata:
  name: test-cluster
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-west-2
userTags:
  adminContact: jdoe
  costCenter: 7536
subnets: 
  - subnet-1
  - subnet-2
  - subnet-3
amiID: ami-96c6f8f7
serviceEndpoints:
  - name: ec2
    url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
hostedZone: Z3URY6TWQ91KVV
1 Required. The installation program prompts you for this value.

2 Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

3 If you do not provide these parameters and values, the installation program provides the default value.

4 The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

5 Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

6 To configure faster storage for etcd, especially for larger clusters, set the storage type as io1 and set iops to 2000.

7 If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

8 The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

9 The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the https protocol and the host must trust the certificate.
The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the `sshKey` value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

Provide the contents of the certificate file that you used for your mirror registry.

Provide the `imageContentSources` section from the output of the command to mirror the repository.

### 2.6.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

- If your cluster is on AWS, you added the ec2.<region>.amazonaws.com, elasticloadbalancing.<region>.amazonaws.com, and s3.<region>.amazonaws.com endpoints to your VPC endpoint. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
  noProxy: example.com  # 3
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  *MY_TRUSTED_CA_CERT*
  -----END CERTIFICATE-----
```

- A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
- A proxy URL to use for creating HTTPS connections outside the cluster.
- A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
- If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

NOTE

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

2.6.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the location of your customized `.install-config.yaml` file.
   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.
The cluster access and credential information also outputs to
<installation_directory>/openshift_install.log when an installation succeeds.

The Ignition config files that the installation program generates contain
certificates that expire after 24 hours, which are then renewed at that time. If
the cluster is shut down before renewing the certificates and the cluster is
later restarted after the 24 hours have elapsed, the cluster automatically
recovers the expired certificates. The exception is that you must manually
approve the pending node-bootstrapper certificate signing requests (CSRs)
to recover kubelet certificates. See the documentation for Recovering from
expired control plane certificates for more information.

It is recommended that you use Ignition config files within 12 hours after they
are generated because the 24-hour certificate rotates from 16 to 22 hours
after the cluster is installed. By using the Ignition config files within 12 hours,
you can avoid installation failure if the certificate update runs during
installation.

You must not delete the installation program or the files that the installation
program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the AdministratorAccess policy from the IAM account that you
used to install the cluster.

The elevated permissions provided by the AdministratorAccess policy are
required only during installation.

2.6.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a
command-line interface. You can install oc on Linux, Windows, or macOS.
2.6.8.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   ```
   $ tar xvzf <file>
   ```
5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```
2.6.8.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

2.6.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```
2.6.10. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the OperatorHub object:

  ```bash
  $ oc patch OperatorHub cluster --type json \n  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
  ```

TIP

Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

2.6.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

2.6.12. Next steps

- Validate an installation.
- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the `must-gather` tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can opt out of remote health reporting.
2.7. INSTALLING A CLUSTER ON AWS INTO AN EXISTING VPC

In OpenShift Container Platform version 4.6, you can install a cluster into an existing Amazon Virtual Private Cloud (VPC) on Amazon Web Services (AWS). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

### 2.7.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- **Configure an AWS account** to host the cluster.

**IMPORTANT**

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see *Managing Access Keys for IAM Users* in the AWS documentation. You can supply the keys when you run the installation program.

- If you use a firewall, you must **configure it to allow the sites** that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can **manually create and maintain IAM credentials**. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

### 2.7.2. About using a custom VPC

In OpenShift Container Platform 4.6, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

### 2.7.2.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
VPC DHCP options

VPC endpoints

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See Amazon VPC console wizard configurations and Work with VPCs and subnets in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See VPC networking components and Route tables for your VPC for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- Create a public and private subnet for each availability zone that your cluster uses. Each availability zone can contain no more than one public and one private subnet. For an example of this type of configuration, see VPC with public and private subnets (NAT) in the AWS documentation.

  Record each subnet ID. Completing the installation requires that you enter these values in the platform section of the install-config.yaml file. See Finding a subnet ID in the AWS documentation.

- The VPC’s CIDR block must contain the Networking.MachineCIDR range, which is the IP address pool for cluster machines. The subnet CIDR blocks must belong to the machine CIDR that you specify.

- The VPC must have a public internet gateway attached to it. For each availability zone:
  - The public subnet requires a route to the internet gateway.
  - The public subnet requires a NAT gateway with an EIP address.
  - The private subnet requires a route to the NAT gateway in public subnet.

- The VPC must not use the kubernetes.io/cluster/.*: owned tag.

  The installation program modifies your subnets to add the kubernetes.io/cluster/.*: shared tag, so your subnets must have at least one free tag slot available for it. See Tag Restrictions in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify.

- You must enable the enableDnsSupport and enableDnsHostnames attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster’s internal DNS records. See DNS Support in Your VPC in the AWS documentation.

  If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the platform.aws.hostedZone field in the install-config.yaml file.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2 and ELB endpoints. To resolve this, you must create a VPC endpoint and attach it to the subnet that the clusters are using. The endpoints should be named as follows:
- ec2.<region>.amazonaws.com
- elasticloadbalancing.<region>.amazonaws.com
- s3.<region>.amazonaws.com

**Required VPC components**

You must provide a suitable VPC and subnets that allow communication to your machines.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td><strong>AWS::EC2::VPC</strong></td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::VPCEndpoint</strong></td>
<td></td>
</tr>
<tr>
<td>Public subnets</td>
<td><strong>AWS::EC2::Subnet</strong></td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::SubnetNetworkAclAssociation</strong></td>
<td></td>
</tr>
<tr>
<td>Internet gateway</td>
<td><strong>AWS::EC2::InternetGateway</strong></td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::VPCGatewayAttachment</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::RouteTable</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::Route</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::SubnetRouteTableAssociation</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::NatGateway</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AWS::EC2::EIP</strong></td>
<td></td>
</tr>
<tr>
<td>Network access</td>
<td><strong>AWS::EC2::NetworkAcl</strong></td>
<td>You must allow the VPC to access the following ports:</td>
</tr>
<tr>
<td>control</td>
<td><strong>AWS::EC2::NetworkAclEntry</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td>Component</td>
<td>AWS type</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td></td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>1024 - 65535</td>
</tr>
<tr>
<td></td>
<td>0 - 65535</td>
</tr>
<tr>
<td>Private subnets</td>
<td>AWS::EC2::Subnet</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::RouteTable</td>
</tr>
<tr>
<td></td>
<td>AWS::EC2::SubnetRouteTableAss ociation</td>
</tr>
</tbody>
</table>

2.7.2.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the `kubernetes.io/cluster/*: shared` tag is removed from the subnets that it used.

2.7.2.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets,
routing tables, Internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

2.7.2.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

2.7.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster. You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

2.7.4. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.
NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   ```
   Agent pid 31874
   ```

   NOTE
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>
   ```

   Example output

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

2.7.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

2.7.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Amazon Web Services (AWS).
Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create the `install-config.yaml` file.
   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>  
   ```

   1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   **IMPORTANT**

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

         **NOTE**

         For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

      ii. Select AWS as the platform to target.

      iii. If you do not have an Amazon Web Services (AWS) profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

      iv. Select the AWS region to deploy the cluster to.

      v. Select the base domain for the Route 53 service that you configured for your cluster.

      vi. Enter a descriptive name for your cluster.

      vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

2.7.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.

NOTE

After installation, you cannot modify these parameters in the install-config.yaml file.

IMPORTANT

The openshift-install command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

2.7.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 2.18. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
</tbody>
</table>
### 2.7.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 2.19. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{.metadata.name}</code>. <code>{.baseDomain}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>.</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{ &quot;auths&quot;: { &quot;cloud.openshift.com&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } }, &quot;quay.io&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } }</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
</tr>
<tr>
<td>values</td>
<td><code>{ &quot;auths&quot;: { &quot;cloud.openshift.com&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } }, &quot;quay.io&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } }</code></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
</tr>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.7.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 2.20. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
### fips

Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
</tbody>
</table>

#### imageContentSources.source

Required if you use `imageContentSources`. Specify the repository that users refer to, for example, in image pull specifications.

- **String**

#### imageContentSources.mirrors

Specify one or more repositories that may also contain the same images.

- **Array of strings**

#### publish

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

- **Internal** or **External**. To deploy a private cluster, which cannot be accessed from the internet, set `publish` to **Internal**. The default value is **External**.
The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```plaintext
sshKey:
  <key1>
  <key2>
  <key3>
```

### 2.7.6.14. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.aws.amiID</code></td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.size</code></td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example <strong>500</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.type</code></td>
<td>The type of the root volume.</td>
<td>Valid AWS EBS volume type, such as <strong>io1</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.type</code></td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid AWS instance type, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>compute.platform.aws.zones</td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a <strong>YAML sequence</strong>.</td>
</tr>
<tr>
<td>compute.aws.region</td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid <strong>AWS region</strong>, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.amiID</td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RH COS AMI.</td>
<td>Any published or custom RH COS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.type</td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid <strong>AWS instance type</strong>, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td>controlPlane.platform.aws.zones</td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a <strong>YAML sequence</strong>.</td>
</tr>
<tr>
<td>controlPlane.aws.region</td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid <strong>AWS region</strong>, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td>platform.aws.amiID</td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RH COS AMI.</td>
<td>Any published or custom RH COS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>platform.aws.serviceEndpoints.name</td>
<td>The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.</td>
<td>Valid <strong>AWS service endpoint</strong> name.</td>
</tr>
</tbody>
</table>
### Parameter | Description | Values
---|---|---
`platform.aws.serviceEndpoints.url` | The AWS service endpoint URL. The URL must use the `https` protocol and the host must trust the certificate. | Valid AWS service endpoint URL.

`platform.aws.userTags` | A map of keys and values that the installation program adds as tags to all resources that it creates. | Any valid YAML map, such as key value pairs in the `<key>: <value>` format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.

`platform.aws.subnets` | If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same `machineNetwork[].cidr` ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone. | Valid subnet IDs.

#### 2.7.6.2. Sample customized install-config.yaml file for AWS

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
credentialsMode: Mint
controlPlane: 
  
hyperthreading: Enabled
name: master
platform:
  aws:
    zones:
    - us-west-2a
    - us-west-2b
    rootVolume:
      iops: 4000
      size: 500
```
Required. The installation program prompts you for this value.

Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

If you do not provide these parameters and values, the installation program provides the default
The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as m4.2xlarge or m5.2xlarge, for your machines if you disable simultaneous multithreading.

To configure faster storage for etcd, especially for larger clusters, set the storage type as io1 and set iops to 2000.

If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the https protocol and the host must trust the certificate.

The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

2.7.6.3. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

- If your cluster is on AWS, you added the `ec2.<region>.amazonaws.com`, `elasticloadbalancing.<region>.amazonaws.com`, and `s3.<region>.amazonaws.com` endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
  noProxy: example.com  # 3
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. A comma-separated list of destination domain names, IP addresses, or other network
A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For

If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster` `Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

2.7.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
...  
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the AdministratorAccess policy from the IAM account that you used to install the cluster.
NOTE
The elevated permissions provided by the AdministratorAccess policy are required only during installation.

2.7.8. Installing the OpenShift CLI by downloading the binary
You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT
If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

2.7.8.1. Installing the OpenShift CLI on Linux
You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure
2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   $ tar xvzf <file>
5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:
   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

2.7.8.2. Installing the OpenShift CLI on Windows
You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure
2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   C:\> oc <command>
   ```

2.7.8.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   $ oc <command>
   ```

2.7.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadm` credentials:
2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

**Example output**

```
system:admin
```

### 2.7.10. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/./openshift_install.log` log file on the installation host.

2. List the OpenShift Container Platform web console route:

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**Example output**

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console
https reencrypt/Redirect None
```
3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the **kubeadmin** user.

**Additional resources**

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

### 2.7.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to **OpenShift Cluster Manager**.

After you confirm that your **OpenShift Cluster Manager** inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, **use subscription watch** to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service.

### 2.7.12. Next steps

- **Validating an installation**.
- **Customize your cluster**.
- If necessary, you can **opt out of remote health reporting**.
- If necessary, you can **remove cloud provider credentials**.

### 2.8. INSTALLING A PRIVATE CLUSTER ON AWS

In OpenShift Container Platform version 4.6, you can install a private cluster into an existing VPC on Amazon Web Services (AWS). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

#### 2.8.1. Prerequisites

- Review details about the **OpenShift Container Platform installation and update** processes.
- **Configure an AWS account** to host the cluster.

**IMPORTANT**

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see [Managing Access Keys for IAM Users](#) in the AWS documentation. You can supply the keys when you run the installation program.
If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

2.8.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the Internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

To deploy a private cluster, you must use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

Additionally, you must deploy a private cluster from a machine that has access the API services for the cloud you provision to, the hosts on the network that you provision, and to the internet to obtain installation media. You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

2.8.2.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for access from only the private network.

The cluster still requires access to Internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the baseDomain for the cluster

The installation program does use the baseDomain that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

2.8.2.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from Internet on 6443 (Kubernetes API port).
If you use a public Service type load balancer, you must tag a public subnet in each availability zone with `kubernetes.io/cluster/<cluster-infra-id>: shared` so that AWS can use them to create public load balancers.

### 2.8.3. About using a custom VPC

In OpenShift Container Platform 4.6, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

#### 2.8.3.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See Amazon VPC console wizard configurations and Work with VPCs and subnets in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See VPC networking components and Route tables for your VPC for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the `kubernetes.io/cluster/.*: owned` tag. The installation program modifies your subnets to add the `kubernetes.io/cluster/.*: shared` tag, so your subnets must have at least one free tag slot available for it. See Tag Restrictions in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify.
• You must enable the `enableDnsSupport` and `enableDnsHostnames` attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster’s internal DNS records. See DNS Support in Your VPC in the AWS documentation. If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the `platform.aws.hostedZone` field in the `install-config.yaml` file.

• If you use a cluster with public access, you must create a public and a private subnet for each availability zone that your cluster uses. Each availability zone can contain no more than one public and one private subnet.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2 and ELB endpoints. To resolve this, you must create a VPC endpoint and attach it to the subnet that the clusters are using. The endpoints should be named as follows:

- `ec2.<region>.amazonaws.com`
- `elasticloadbalancing.<region>.amazonaws.com`
- `s3.<region>.amazonaws.com`

**Required VPC components**

You must provide a suitable VPC and subnets that allow communication to your machines.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VPC</strong></td>
<td><code>AWS::EC2::VPC</code></td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::VPCEndpoint</code></td>
<td></td>
</tr>
<tr>
<td><strong>Public subnets</strong></td>
<td><code>AWS::EC2::Subnet</code></td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::SubnetNetworkAclAssociation</code></td>
<td></td>
</tr>
<tr>
<td><strong>Internet gateway</strong></td>
<td><code>AWS::EC2::InternetGateway</code></td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::VPCEndpoint</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::VPCGatewayAttachment</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::RouteTable</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::Route</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::SubnetRouteTableAssociation</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::NatGateway</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>AWS::EC2::EIP</code></td>
<td></td>
</tr>
</tbody>
</table>
## Network access control

- **AWS::EC2::NetworkAcl**
- **AWS::EC2::NetworkAclEntry**

You must allow the VPC to access the following ports:

<table>
<thead>
<tr>
<th>Port</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td>1024 - 65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td>0 - 65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
</tbody>
</table>

### Private subnets

- **AWS::EC2::Subnet**
- **AWS::EC2::RouteTable**
- **AWS::EC2::SubnetRouteTableAssociation**

Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

## 2.8.3.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.
If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the
OpenShift Container Platform cluster from a VPC, the `kubernetes.io/cluster/*: shared` tag is removed
from the subnets that it used.

### 2.8.3.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required
for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics
the division of permissions that you might have at your company: some individuals can create different
resource in your clouds than others. For example, you might be able to create application-specific items,
like instances, buckets, and load balancers, but not networking-related components such as VPCs,
subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions
that are required to make VPCs and core networking components within the VPC, such as subnets,
routing tables, Internet gateways, NAT, and VPN. You still need permission to make the application
resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and
nodes.

### 2.8.3.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is
reduced in the following ways:

- You can install multiple OpenShift Container Platform clusters in the same VPC.
- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

### 2.8.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform
  subscription management. If the cluster has internet access and you do not disable Telemetry,
  that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

2.8.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:
$ eval "$(ssh-agent -s)"

**Example output**

Agent pid 31874

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add /<path>/<file_name>
   ```

   **Example output**

   

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 2.8.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.
IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

2.8.7. Manually creating the installation configuration file

For installations of a private OpenShift Container Platform cluster that are only accessible from an internal network and are not visible to the Internet, you must manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```bash
   $ mkdir <installation_directory>
   ```

   IMPORTANT

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   NOTE

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
IMPORTANT

The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

2.8.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

2.8.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 2.22. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (‐), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{  &quot;auths&quot;:{  &quot;cloud.openshift.com&quot;:{  &quot;auth&quot;:&quot;b3Blb=&quot;,  &quot;email&quot;:&quot;you@example.com&quot;  },  &quot;quay.io&quot;:{  &quot;auth&quot;:&quot;b3Blb=&quot;,  &quot;email&quot;:&quot;you@example.com&quot;  }  }}</code></td>
</tr>
</tbody>
</table>

### 2.8.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 2.23. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>

210
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>10.128.0.0/14</code> with a host prefix of <code>/23</code>.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>cidr: 10.128.0.0/14</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>hostPrefix: 23</code></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between <code>0</code> and <code>32</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An IPv4 network.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is <code>23</code>.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to <code>23</code> then each node is assigned a <code>/23</code> subnet out of the given cidr. A hostPrefix value of <code>23</code> provides <code>510</code> (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is <code>23</code>.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <code>172.30.0.0/16</code>.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>172.30.0.0/16</code></td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>10.0.0.0/16</code></td>
</tr>
</tbody>
</table>
networking.machine Network.cidr

Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.

An IP network block in CIDR notation. For example, 10.0.0.0/16.

NOTE

Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

2.8.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 2.24. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hypertreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the <code>x86_64</code> architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specifying one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
The SSH key or keys to authenticate access your cluster machines.

**NOTE**
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: <em>sshKey:</em> <code>&lt;key1&gt;</code> <code>&lt;key2&gt;</code> <code>&lt;key3&gt;</code></td>
</tr>
</tbody>
</table>

### 2.8.7.1.4. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

**Table 2.25. Optional AWS parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.aws.amiID</td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.iops</td>
<td>The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.</td>
<td>Integer, for example <strong>4000</strong>.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.size</td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example <strong>500</strong>.</td>
</tr>
<tr>
<td>compute.platform.aws.rootVolume.type</td>
<td>The type of the root volume.</td>
<td>Valid <strong>AWS EBS volume type</strong>, such as <strong>io1</strong>.</td>
</tr>
<tr>
<td>compute.platform.aws.type</td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid <strong>AWS instance type</strong>, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>compute.platform.aws.zones</strong></td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><strong>compute.aws.region</strong></td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.amiID</strong></td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.type</strong></td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid AWS instance type, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.aws.zones</strong></td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <strong>us-east-1c</strong>, in a YAML sequence.</td>
</tr>
<tr>
<td><strong>controlPlane.aws.region</strong></td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid AWS region, such as <strong>us-east-1</strong>.</td>
</tr>
<tr>
<td><strong>platform.aws.amiID</strong></td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><strong>platform.aws.serviceEndpoints.name</strong></td>
<td>The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.</td>
<td>Valid AWS service endpoint name.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.aws.serviceEndpoints.url</td>
<td>The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate.</td>
<td>Valid AWS service endpoint URL.</td>
</tr>
<tr>
<td>platform.aws.userTags</td>
<td>A map of keys and values that the installation program adds as tags to all resources that it creates.</td>
<td>Any valid YAML map, such as key value pairs in the &lt;key&gt;: &lt;value&gt; format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation.</td>
</tr>
<tr>
<td>platform.aws.subnets</td>
<td>If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone.</td>
<td>Valid subnet IDs.</td>
</tr>
</tbody>
</table>

2.8.7.2. Sample customized install-config.yaml file for AWS

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your install-config.yaml file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
credentialsMode: Mint
controlPlane:
  hyperthreading: Enabled
name: master
platform:
  aws:
    zones:
    - us-west-2a
    - us-west-2b
    rootVolume:
      iops: 4000
      size: 500
```
Required. The installation program prompts you for this value.

Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified mode, instead of having the CCO dynamically try to determine the capabilities of the credentials. For details about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as m4.2xlarge or m5.2xlarge, for your machines if you disable simultaneous multithreading.

To configure faster storage for etcd, especially for larger clusters, set the storage type as io1 and set iops to 2000.

If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the https protocol and the host must trust the certificate.

The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.
How to publish the user-facing endpoints of your cluster. Set publish to Internal to deploy a private cluster, which cannot be accessed from the Internet. The default value is External.

2.8.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the install-config.yaml file.

Prerequisites

- You have an existing install-config.yaml file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

**NOTE**

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

- If your cluster is on AWS, you added the ec2.<region>.amazonaws.com, elasticloadbalancing.<region>.amazonaws.com, and s3.<region>.amazonaws.com endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...
```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.

If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**
Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 2.8.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**
You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**
1. Change to the directory that contains the installation program and initialize the cluster deployment:

```bash
$ ./openshift-install create cluster --dir <installation_directory> \  
  --log-level=info
```

1. For `<installation_directory>`, specify the
2. To view different installation details, specify `warn, debug,` or `error` instead of `info`.

---

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```bash
...  
INFO Install complete!  
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'  
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com  
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"  
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
2.8.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of `oc`, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of `oc`.

2.8.9.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Linux Client** entry and save the file.
4. Unpack the archive:
   
   ```
   $ tar xvzf <file>
   ```
5. Place the `oc` binary in a directory that is on your **PATH**.
   
   To check your **PATH**, execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

2.8.9.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   ```
   C:> path
   ```

   After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   C:> oc <command>
   ```

### 2.8.9.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   $ oc <command>
   ```

### 2.8.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```bash
$ oc whoami
```

Example output

```
system:admin
```

2.8.11. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

```bash
$ cat <installation_directory>/auth/kubeadmin-password
```

**NOTE**

Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/.openshift_install.log` log file on the installation host.

2. List the OpenShift Container Platform web console route:

```bash
$ oc get routes -n openshift-console | grep 'console-openshift'
```

**NOTE**

Alternatively, you can obtain the OpenShift Container Platform route from the `<installation_directory>/.openshift_install.log` log file on the installation host.

Example output

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console https reencrypt/Redirect None
```
3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the kubeadmin user.

Additional resources
- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

2.8.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources
- See About remote health monitoring for more information about the Telemetry service.

2.8.13. Next steps
- Validating an installation.
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If necessary, you can remove cloud provider credentials.

2.9. INSTALLING A CLUSTER ON AWS INTO A GOVERNMENT REGION

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) into a government region. To configure the government region, modify parameters in the install-config.yaml file before you install the cluster.

2.9.1. Prerequisites
- Review details about the OpenShift Container Platform installation and update processes.
- Configure an AWS account to host the cluster.

IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use long-lived credentials. To generate appropriate keys, see Managing Access Keys for IAM Users in the AWS documentation. You can supply the keys when you run the installation program.
If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

2.9.2. AWS government regions

OpenShift Container Platform supports deploying a cluster to AWS GovCloud (US) regions. AWS GovCloud is specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads in the cloud.

These regions do not have published Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Images (AMI) to select, so you must upload a custom AMI that belongs to that region.

The following AWS GovCloud partitions are supported:

- us-gov-west-1
- us-gov-east-1

The AWS GovCloud region and custom AMI must be manually configured in the `install-config.yaml` file since RHCOS AMIs are not provided by Red Hat for those regions.

2.9.3. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the Internet.

**NOTE**

Public zones are not supported in Route 53 in AWS GovCloud. Therefore, clusters must be private if they are deployed to an AWS government region.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

To deploy a private cluster, you must use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

Additionally, you must deploy a private cluster from a machine that has access the API services for the cloud you provision to, the hosts on the network that you provision, and to the internet to obtain installation media. You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

2.9.3.1. Private clusters in AWS

To create a private cluster on Amazon Web Services (AWS), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for
access from only the private network.

The cluster still requires access to Internet to access the AWS APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public load balancers, which support public ingress
- A public Route 53 zone that matches the baseDomain for the cluster

The installation program does use the baseDomain that you specify to create a private Route 53 zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

2.9.3.1.1. Limitations

The ability to add public functionality to a private cluster is limited.

- You cannot make the Kubernetes API endpoints public after installation without taking additional actions, including creating public subnets in the VPC for each availability zone in use, creating a public load balancer, and configuring the control plane security groups to allow traffic from Internet on 6443 (Kubernetes API port).

- If you use a public Service type load balancer, you must tag a public subnet in each availability zone with kubernetes.io/cluster/<cluster-infra-id>: shared so that AWS can use them to create public load balancers.

2.9.4. About using a custom VPC

In OpenShift Container Platform 4.6, you can deploy a cluster into existing subnets in an existing Amazon Virtual Private Cloud (VPC) in Amazon Web Services (AWS). By deploying OpenShift Container Platform into an existing AWS VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. If you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself, use this installation option.

Because the installation program cannot know what other components are also in your existing subnets, it cannot choose subnet CIDRs and so forth on your behalf. You must configure networking for the subnets that you install your cluster to yourself.

2.9.4.1. Requirements for using your VPC

The installation program no longer creates the following components:

- Internet gateways
- NAT gateways
- Subnets
- Route tables
- VPCs
- VPC DHCP options
- VPC endpoints

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. See Amazon VPC console wizard configurations and Work with VPCs and subnets in the AWS documentation for more information on creating and managing an AWS VPC.

The installation program cannot:

- Subdivide network ranges for the cluster to use.
- Set route tables for the subnets.
- Set VPC options like DHCP.

You must complete these tasks before you install the cluster. See VPC networking components and Route tables for your VPC for more information on configuring networking in an AWS VPC.

Your VPC must meet the following characteristics:

- The VPC must not use the kubernetes.io/cluster/.*: owned tag. The installation program modifies your subnets to add the kubernetes.io/cluster/.*: shared tag, so your subnets must have at least one free tag slot available for it. See Tag Restrictions in the AWS documentation to confirm that the installation program can add a tag to each subnet that you specify.

- You must enable the enableDnsSupport and enableDnsHostnames attributes in your VPC, so that the cluster can use the Route 53 zones that are attached to the VPC to resolve cluster’s internal DNS records. See DNS Support in Your VPC in the AWS documentation.

- If you prefer to use your own Route 53 hosted private zone, you must associate the existing hosted zone with your VPC prior to installing a cluster. You can define your hosted zone using the platform.aws.hostedZone field in the install-config.yaml file.

- If you use a cluster with public access, you must create a public and a private subnet for each availability zone that your cluster uses. Each availability zone can contain no more than one public and one private subnet.

If you are working in a disconnected environment, you are unable to reach the public IP addresses for EC2 and ELB endpoints. To resolve this, you must create a VPC endpoint and attach it to the subnet that the clusters are using. The endpoints should be named as follows:

- ec2.<region>.amazonaws.com
- elasticloadbalancing.<region>.amazonaws.com
- s3.<region>.amazonaws.com

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.
<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td>• AWS::EC2::VPC</td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::VPCEndpoint</td>
<td></td>
</tr>
<tr>
<td>Public subnets</td>
<td>• AWS::EC2::Subnet</td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::SubnetNetworkAclAssociation</td>
<td></td>
</tr>
<tr>
<td>Internet gateway</td>
<td>• AWS::EC2::InternetGateway</td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::VPCGatewayAttachment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::RouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::Route</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::SubnetRouteTableAssociation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::NatGateway</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::EIP</td>
<td></td>
</tr>
<tr>
<td>Network access</td>
<td>• AWS::EC2::NetworkAcl</td>
<td>You must allow the VPC to access the following ports:</td>
</tr>
<tr>
<td>control</td>
<td>• AWS::EC2::NetworkAclEntry</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Port</strong></td>
<td><strong>Reason</strong></td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td></td>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td></td>
<td>1024 - 65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td></td>
<td>0 - 65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
</tbody>
</table>
Private subnets

- AWS::EC2::Subnet
- AWS::EC2::RouteTable
- AWS::EC2::SubnetRouteTableAssociation

Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

2.9.4.2. VPC validation

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist.
- You provide private subnets.
- The subnet CIDRs belong to the machine CIDR that you specified.
- You provide subnets for each availability zone. Each availability zone contains no more than one public and one private subnet. If you use a private cluster, provide only a private subnet for each availability zone. Otherwise, provide exactly one public and private subnet for each availability zone.
- You provide a public subnet for each private subnet availability zone. Machines are not provisioned in availability zones that you do not provide private subnets for.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted. When you remove the OpenShift Container Platform cluster from a VPC, the kubernetes.io/cluster/.*: shared tag is removed from the subnets that it used.

2.9.4.3. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resource in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or ingress rules.

The AWS credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, Internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as ELBs, security groups, S3 buckets, and nodes.

2.9.4.4. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is reduced in the following ways:
You can install multiple OpenShift Container Platform clusters in the same VPC.

- ICMP ingress is allowed from the entire network.
- TCP 22 ingress (SSH) is allowed to the entire network.
- Control plane TCP 6443 ingress (Kubernetes API) is allowed to the entire network.
- Control plane TCP 22623 ingress (MCS) is allowed to the entire network.

2.9.5. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

2.9.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N " \n
1 Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

NOTE
If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```
$ eval "$(ssh-agent -s)"

Example output

Agent pid 31874

NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name>

Example output

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

2.9.7. Obtaining the installation program
Before you install OpenShift Container Platform, download the installation file on a local computer.

### Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

### Procedure

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 2.9.8. Manually creating the installation configuration file

When installing OpenShift Container Platform on Amazon Web Services (AWS) into a region requiring a custom Red Hat Enterprise Linux CoreOS (RHCOS) AMI, you must manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**
1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

**NOTE**

You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 2.9.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 2.9.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<p>| Table 2.26. Required parameters |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <em>install-config.yaml</em> content. The current version is <strong>v1</strong>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <em>ObjectMeta</em>, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 2.9.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 2.27. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given <code>cidr</code>. A <code>hostPrefix</code> value of 23 provides 510 (2^{(32 - 23)} - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
</tbody>
</table>
| networking.serviceNetwork | The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network. | An array with an IP address block in CIDR format. For example: `networking:
  serviceNetwork:
  - 172.30.0.0/16` |
| networking.machineNetwork | The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. | An array of objects. For example: `networking:
  machineNetwork:
  - cidr: 10.0.0.0/16` |
| networking.machineNetwork.cidr | Required if you use `networking.machineNetwork`. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24. | An IP network block in CIDR notation. For example, 10.0.0.0/16. **NOTE** Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in. |

2.9.8.13. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 2.28. Optional parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <strong>compute</strong>. The name of the machine pool.</td>
<td><strong>worker</strong></td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <strong>compute</strong>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <strong>controlPlane.platform</strong> parameter value.</td>
<td><strong>aws</strong>, <strong>azure</strong>, <strong>gcp</strong>, <strong>openstack</strong>, <strong>ovirt</strong>, <strong>vsphere</strong>, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <strong>MachinePool</strong> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architect</td>
<td>Determines the instruction set architecture of the machines in the pool.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Currently, heterogeneous clusters are not supported, so all pools must</td>
<td></td>
</tr>
<tr>
<td></td>
<td>specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td></td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or **hyperv</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td>threading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere, or</strong>{}</td>
</tr>
<tr>
<td></td>
<td>provider that hosts the control plane machines. This parameter value must</td>
<td></td>
</tr>
<tr>
<td></td>
<td>match the <strong>compute.platform</strong> parameter value.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (“”).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
Enable or disable FIPS mode. The default is **false** (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

### imageContentSources

Sources and repositories for the release-image content.

- **Array of objects.** Includes a **source** and, optionally, **mirrors**, as described in the following rows of this table.

#### imageContentSources.source

Required if you use **imageContentSources**. Specify the repository that users refer to, for example, in image pull specifications.

- **String**

#### imageContentSources.mirrors

Specify one or more repositories that may also contain the same images.

- **Array of strings**

#### publish

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

- **Internal** or **External**. To deploy a private cluster, which cannot be accessed from the internet, set **publish** to **Internal**. The default value is **External**.
The SSH key or keys to authenticate access your cluster machines.

**NOTE**
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 2.9.8.1.4. Optional AWS configuration parameters

Optional AWS configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.amiID</code></td>
<td>The AWS AMI used to boot compute machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.iops</code></td>
<td>The Input/Output Operations Per Second (IOPS) that is reserved for the root volume.</td>
<td>Integer, for example <strong>4000</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.size</code></td>
<td>The size in GiB of the root volume.</td>
<td>Integer, for example <strong>500</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.rootVolume.type</code></td>
<td>The type of the root volume.</td>
<td>Valid <a href="#">AWS EBS volume type</a>, such as <strong>io1</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.aws.type</code></td>
<td>The EC2 instance type for the compute machines.</td>
<td>Valid <a href="#">AWS instance type</a>, such as <strong>c5.9xlarge</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>compute.platform.aws.zones</code></td>
<td>The availability zones where the installation program creates machines for the compute machine pool. If you provide your own VPC, you must provide a subnet in that availability zone.</td>
<td>A list of valid AWS availability zones, such as <code>us-east-1c</code>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>compute.aws.region</code></td>
<td>The AWS region that the installation program creates compute resources in.</td>
<td>Any valid AWS region, such as <code>us-east-1</code>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.amiID</code></td>
<td>The AWS AMI used to boot control plane machines for the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.type</code></td>
<td>The EC2 instance type for the control plane machines.</td>
<td>Valid AWS instance type, such as <code>c5.9xlarge</code>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.aws.zones</code></td>
<td>The availability zones where the installation program creates machines for the control plane machine pool.</td>
<td>A list of valid AWS availability zones, such as <code>us-east-1c</code>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>controlPlane.aws.region</code></td>
<td>The AWS region that the installation program creates control plane resources in.</td>
<td>Valid AWS region, such as <code>us-east-1</code>.</td>
</tr>
<tr>
<td><code>platform.aws.amiID</code></td>
<td>The AWS AMI used to boot all machines for the cluster. If set, the AMI must belong to the same region as the cluster. This is required for regions that require a custom RHCOS AMI.</td>
<td>Any published or custom RHCOS AMI that belongs to the set AWS region.</td>
</tr>
<tr>
<td><code>platform.aws.serviceEndpoints.name</code></td>
<td>The AWS service endpoint name. Custom endpoints are only required for cases where alternative AWS endpoints, like FIPS, must be used. Custom API endpoints can be specified for EC2, S3, IAM, Elastic Load Balancing, Tagging, Route 53, and STS AWS services.</td>
<td>Valid AWS service endpoint name.</td>
</tr>
</tbody>
</table>
 platform.aws.serviceEndpoints.url | The AWS service endpoint URL. The URL must use the https protocol and the host must trust the certificate. | Valid AWS service endpoint URL. |
platform.aws.userTags | A map of keys and values that the installation program adds as tags to all resources that it creates. | Any valid YAML map, such as key value pairs in the <key>: <value> format. For more information about AWS tags, see Tagging Your Amazon EC2 Resources in the AWS documentation. |
platform.aws.subnets | If you provide the VPC instead of allowing the installation program to create the VPC for you, specify the subnet for the cluster to use. The subnet must be part of the same machineNetwork[].cidr ranges that you specify. For a standard cluster, specify a public and a private subnet for each availability zone. For a private cluster, specify a private subnet for each availability zone. | Valid subnet IDs. |

### 2.9.8.2. Sample customized install-config.yaml file for AWS

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your install-config.yaml file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com  
credentialsMode: Mint  
controlPlane:  
hyperthreading: Enabled  
name: master  
platform:  
aws:  
  zones:  
    - us-gov-west-1a  
    - us-gov-west-1b  
rootVolume:  
iops: 4000  
size: 500  
```
Required.

Optional: Add this parameter to force the Cloud Credential Operator (CCO) to use the specified
If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger instance types, such as m4.2xlarge or m5.2xlarge, for your machines if you disable simultaneous multithreading.

To configure faster storage for etcd, especially for larger clusters, set the storage type as io1 and set iops to 2000.

If you provide your own VPC, specify subnets for each availability zone that your cluster uses.

The ID of the AMI used to boot machines for the cluster. If set, the AMI must belong to the same region as the cluster.

The AWS service endpoints. Custom endpoints are required when installing to an unknown AWS region. The endpoint URL must use the https protocol and the host must trust the certificate.

The ID of your existing Route 53 private hosted zone. Providing an existing hosted zone requires that you supply your own VPC and the hosted zone is already associated with the VPC prior to installing your cluster. If undefined, the installation program creates a new hosted zone.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.
2.9.8.3. AWS regions without a published RHCOS AMI

You can deploy an OpenShift Container Platform cluster to Amazon Web Services (AWS) regions without native support for a Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) or the AWS software development kit (SDK). If a published AMI is not available for an AWS region, you can upload a custom AMI prior to installing the cluster. This is required if you are deploying your cluster to an AWS government region.

If you are deploying to a non-government region that does not have a published RHCOS AMI, and you do not specify a custom AMI, the installation program copies the us-east-1 AMI to the user account automatically. Then the installation program creates the control plane machines with encrypted EBS volumes using the default or user-specified Key Management Service (KMS) key. This allows the AMI to follow the same process workflow as published RHCOS AMIs.

A region without native support for an RHCOS AMI is not available to select from the terminal during cluster creation because it is not published. However, you can install to this region by configuring the custom AMI in the install-config.yaml file.

2.9.8.4. Uploading a custom RHCOS AMI in AWS

If you are deploying to a custom Amazon Web Services (AWS) region, you must upload a custom Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) that belongs to that region.

Prerequisites

- You configured an AWS account.
- You created an Amazon S3 bucket with the required IAM service role.
- You uploaded your RHCOS VMDK file to Amazon S3. The RHCOS VMDK file must be the highest version that is less than or equal to the OpenShift Container Platform version you are installing.
- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer.

Procedure

1. Export your AWS profile as an environment variable:

   $ export AWS_PROFILE=<aws_profile>

   The AWS profile name that holds your AWS credentials, like govcloud.

2. Export the region to associate with your custom AMI as an environment variable:

   $ export AWS_DEFAULT_REGION=<aws_region>
The AWS region, like **us-gov-east-1**.

3. Export the version of RHCOS you uploaded to Amazon S3 as an environment variable:

   ```bash
   $ export RHCOS_VERSION=<version>
   ```

   The RHCOS VMDK version, like **4.6.0**.

4. Export the Amazon S3 bucket name as an environment variable:

   ```bash
   $ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
   ```

5. Create the **containers.json** file and define your RHCOS VMDK file:

   ```bash
   $ cat <<EOF > containers.json
   {
     "Description": "rhcos-$\{RHCOS_VERSION\}-x86_64-aws.x86_64",
     "Format": "vmdk",
     "UserBucket": {
       "S3Bucket": "$\{VMIMPORT_BUCKET_NAME\}",
       "S3Key": "rhcos-$\{RHCOS_VERSION\}-x86_64-aws.x86_64.vmdk"
     }
   }
   EOF
   ```

6. Import the RHCOS disk as an Amazon EBS snapshot:

   ```bash
   $ aws ec2 import-snapshot --region $\{AWS_DEFAULT_REGION\} \
     --description "<description>" \
     --disk-container "file://<file_path>/containers.json"
   ```

   The description of your RHCOS disk being imported, like **rhcos-$\{RHCOS_VERSION\}-x86_64-aws.x86_64**.

   The file path to the JSON file describing your RHCOS disk. The JSON file should contain your Amazon S3 bucket name and key.

7. Check the status of the image import:

   ```bash
   $ watch -n 5 aws ec2 describe-import-snapshot-tasks --region $\{AWS_DEFAULT_REGION\}
   ```

   **Example output**

   ```json
   {
     "ImportSnapshotTasks": [
       {
         "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
         "ImportTaskId": "import-snap-fh6i8uil",
         "SnapshotTaskDetail": {
           "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
           "DiskImageSize": 819056640.0,
         }
       }
     ]
   }
   ```
Copy the **SnapshotId** to register the image.

8. Create a custom RHCOS AMI from the RHCOS snapshot:

```bash
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs={DeleteOnTermination=true,SnapshotId=<snapshot_ID>}'
```

1. The RHCOS VMDK architecture type, like **x86_64**, **s390x**, or **ppc64le**.
2. The **Description** from the imported snapshot.
3. The name of the RHCOS AMI.
4. The **SnapshotId** from the imported snapshot.

To learn more about these APIs, see the AWS documentation for importing snapshots and creating EBS-backed AMIs.

### 2.9.8.5. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing **install-config.yaml** file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object’s **spec.noProxy** field to bypass the proxy if necessary.
NOTE

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

- If your cluster is on AWS, you added the `ec2.<region>.amazonaws.com`, `elasticloadbalancing.<region>.amazonaws.com`, and `s3.<region>.amazonaws.com` endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>
     httpsProxy: https://<username>:<pswd>@<ip>:<port>
     noProxy: example.com
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   
   ^1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   ^2 A proxy URL to use for creating HTTPS connections outside the cluster.

   ^3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

   ^4 If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.
NOTE
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

NOTE
Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

2.9.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   NOTE
   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.
NOTE

The cluster access and credential information also outputs to
<installation_directory>/openshift_install.log when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

2. Optional: Remove or disable the AdministratorAccess policy from the IAM account that you used to install the cluster.

NOTE

The elevated permissions provided by the AdministratorAccess policy are required only during installation.

2.9.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

2.9.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

2.9.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
2.9.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```bash
$ oc <command>
```

2.9.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```
2.9.12. Logging in to the cluster by using the web console

The `kubeadmin` user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the `kubeadmin` user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the `kubeadmin` user from the `kubeadmin-password` file on the installation host:

   ```bash
   $ cat <installation_directory>/auth/kubeadmin-password
   ```

   **NOTE**
   
   Alternatively, you can obtain the `kubeadmin` password from the `<installation_directory>/openshift_install.log` log file on the installation host.

2. List the OpenShift Container Platform web console route:

   ```bash
   $ oc get routes -n openshift-console | grep 'console-openshift'
   ```

   **NOTE**
   
   Alternatively, you can obtain the OpenShift Container Platform route from the `<installation_directory>/openshift_install.log` log file on the installation host.

   **Example output**

   ```
   console     console-openshift-console.apps.<cluster_name>.<base_domain>            console
   https       reencrypt/Redirect   None
   ```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the `kubeadmin` user.

**Additional resources**

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

2.9.13. Telemetry access for OpenShift Container Platform
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

2.9.14. Next steps

- Validating an installation.
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If necessary, you can remove cloud provider credentials.

2.10. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN AWS BY USING CLOUDFORMATION TEMPLATES

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) that uses infrastructure that you provide.

One way to create this infrastructure is to use the provided CloudFormation templates. You can modify the templates to customize your infrastructure or use the information that they contain to create AWS objects according to your company’s policies.

IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several CloudFormation templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

2.10.1. Prerequisites

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You configured an AWS account to host the cluster.
**IMPORTANT**

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see Managing Access Keys for IAM Users in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix) in the AWS documentation.

- If you use a firewall, you configured it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

### 2.10.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 2.10.3. Required AWS infrastructure components

To install OpenShift Container Platform on user-provisioned infrastructure in Amazon Web Services (AWS), you must manually create both the machines and their supporting infrastructure.

For more information about the integration testing for different platforms, see the OpenShift Container Platform 4.x Tested Integrations page.
By using the provided CloudFormation templates, you can create stacks of AWS resources that represent the following components:

- An AWS Virtual Private Cloud (VPC)
- Networking and load balancing components
- Security groups and roles
- An OpenShift Container Platform bootstrap node
- OpenShift Container Platform control plane nodes
- An OpenShift Container Platform compute node

Alternatively, you can manually create the components or you can reuse existing infrastructure that meets the cluster requirements. Review the CloudFormation templates for more details about how the components interrelate.

### 2.10.3.1. Cluster machines

You need **AWS::EC2::Instance** objects for the following machines:

- A bootstrap machine. This machine is required during installation, but you can remove it after your cluster deploys.
- Three control plane machines. The control plane machines are not governed by a machine set.
- Compute machines. You must create at least two compute machines, which are also known as worker machines, during installation. These machines are not governed by a machine set.

You can use the following instance types for the cluster machines with the provided CloudFormation templates.

**IMPORTANT**

If `m4` instance types are not available in your region, such as with `eu-west-3`, use `m5` types instead.

#### Table 2.30. Instance types for machines

<table>
<thead>
<tr>
<th>Instance type</th>
<th>Bootstrap</th>
<th>Control plane</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>i3.large</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m4.large</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>m4.xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.2xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.4xlarge</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Instance type</td>
<td>Bootstrap</td>
<td>Control plane</td>
<td>Compute</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>m4.8xlarge</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>m4.10xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m4.16xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m5.large</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>m5.xlarge</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>m5.2xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m5.4xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m5.8xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m5.10xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m5.16xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m6i.xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c4.2xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c4.4xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c4.8xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r4.large</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>r4.xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r4.2xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r4.4xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r4.8xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r4.16xlarge</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

You might be able to use other instance types that meet the specifications of these instance types.

2.10.3.2. Other infrastructure components

- A VPC
• DNS entries
• Load balancers (classic or network) and listeners
• A public and a private Route 53 zone
• Security groups
• IAM roles
• S3 buckets

If you are working in a disconnected environment or use a proxy, you cannot reach the public IP addresses for EC2 and ELB endpoints. To reach these endpoints, you must create a VPC endpoint and attach it to the subnet that the clusters are using. Create the following endpoints:

• ec2.<region>.amazonaws.com
• elasticloadbalancing.<region>.amazonaws.com
• s3.<region>.amazonaws.com

Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td>• AWS::EC2::VPC</td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::VPCEndpoint</td>
<td></td>
</tr>
<tr>
<td>Public subnets</td>
<td>• AWS::EC2::Subnet</td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::SubnetNetworkAclAssociation</td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>AWS type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Internet gateway</td>
<td>• AWS::EC2::InternetGateway</td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::VPCGatewayAttachment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::RouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::Route</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::SubnetRouteTableAssociation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::NatGateway</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::EIP</td>
<td></td>
</tr>
<tr>
<td>Network access control</td>
<td>• AWS::EC2::NetworkAcl</td>
<td>You must allow the VPC to access the following ports:</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::NetworkAclEntry</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port</td>
<td>Reason</td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td></td>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td></td>
<td>1024 - 65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td></td>
<td>0 - 65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
<tr>
<td>Private subnets</td>
<td>• AWS::EC2::Subnet</td>
<td>Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.</td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::RouteTable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AWS::EC2::SubnetRouteTableAssociation</td>
<td></td>
</tr>
</tbody>
</table>

**Required DNS and load balancing components**

Your DNS and load balancer configuration needs to use a public hosted zone and can use a private hosted zone similar to the one that the installation program uses if it provisions the cluster’s infrastructure. You must create a DNS entry that resolves to your load balancer. An entry for `api`. |
<cluster_name>.<domain> must point to the external load balancer, and an entry for api-int. <cluster_name>.<domain> must point to the internal load balancer.

The cluster also requires load balancers and listeners for port 6443, which are required for the Kubernetes API and its extensions, and port 22623, which are required for the Ignition config files for new machines. The targets will be the control plane nodes (also known as the master nodes). Port 6443 must be accessible to both clients external to the cluster and nodes within the cluster. Port 22623 must be accessible to nodes within the cluster.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>AWS::Route 53::HostedZone</td>
<td>The hosted zone for your internal DNS.</td>
</tr>
<tr>
<td>etcd record sets</td>
<td>AWS::Route 53::RecordSet</td>
<td>The registration records for etcd for your control plane machines.</td>
</tr>
<tr>
<td>Public load balancer</td>
<td>AWS::ElasticLoadBalancingV2::LoadBalancer</td>
<td>The load balancer for your public subnets.</td>
</tr>
<tr>
<td>External API server record</td>
<td>AWS::Route 53::RecordSetGroup</td>
<td>Alias records for the external API server.</td>
</tr>
<tr>
<td>External listener</td>
<td>AWS::ElasticLoadBalancingV2::Listener</td>
<td>A listener on port 6443 for the external load balancer.</td>
</tr>
<tr>
<td>External target group</td>
<td>AWS::ElasticLoadBalancingV2::TargetGroup</td>
<td>The target group for the external load balancer.</td>
</tr>
<tr>
<td>Private load balancer</td>
<td>AWS::ElasticLoadBalancingV2::LoadBalancer</td>
<td>The load balancer for your private subnets.</td>
</tr>
<tr>
<td>Internal API server record</td>
<td>AWS::Route 53::RecordSetGroup</td>
<td>Alias records for the internal API server.</td>
</tr>
<tr>
<td>Internal listener</td>
<td>AWS::ElasticLoadBalancingV2::Listener</td>
<td>A listener on port 22623 for the internal load balancer.</td>
</tr>
<tr>
<td>Component</td>
<td>AWS type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Internal target group</td>
<td>AWS::ElasticLoadBalancingV2::TargetGroup</td>
<td>The target group for the internal load balancer.</td>
</tr>
<tr>
<td>Internal listener</td>
<td>AWS::ElasticLoadBalancingV2::Listener</td>
<td>A listener on port 6443 for the internal load balancer.</td>
</tr>
<tr>
<td>Internal target group</td>
<td>AWS::ElasticLoadBalancingV2::TargetGroup</td>
<td>The target group for the internal load balancer.</td>
</tr>
</tbody>
</table>

### Security groups

The control plane and worker machines require access to the following ports:

<table>
<thead>
<tr>
<th>Group</th>
<th>Type</th>
<th>IP Protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterSecurityGroup</td>
<td>AWS::EC2::SecurityGroup</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>6443</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22623</td>
</tr>
<tr>
<td>WorkerSecurityGroup</td>
<td>AWS::EC2::SecurityGroup</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td>BootstrapSecurityGroup</td>
<td>AWS::EC2::SecurityGroup</td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>19531</td>
</tr>
</tbody>
</table>

### Control plane Ingress

The control plane machines require the following Ingress groups. Each Ingress group is a AWS::EC2::SecurityGroupIngress resource.

<table>
<thead>
<tr>
<th>Ingress group</th>
<th>Description</th>
<th>IP protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterIngress</td>
<td>etcd</td>
<td>tcp</td>
<td>2379–2380</td>
</tr>
<tr>
<td>Ingress group</td>
<td>Description</td>
<td>IP protocol</td>
<td>Port range</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>MasterIngress Vxlan</td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td>MasterIngress WorkerVxlan</td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td>MasterIngress Internal</td>
<td>Internal cluster communication and Kubernetes proxy metrics</td>
<td>tcp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>MasterIngress WorkerInternal</td>
<td>Internal cluster communication</td>
<td>tcp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>MasterIngress Kube</td>
<td>Kubernetes kubelet, scheduler and controller manager</td>
<td>tcp</td>
<td>10250 - 10259</td>
</tr>
<tr>
<td>MasterIngress WorkerKube</td>
<td>Kubernetes kubelet, scheduler and controller manager</td>
<td>tcp</td>
<td>10250 - 10259</td>
</tr>
<tr>
<td>MasterIngress IngressServices</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress WorkerIngress Services</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress Geneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>MasterIngress WorkerGeneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>MasterIngress IpsecIke</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>MasterIngress WorkerIpsecIke</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>MasterIngress IpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>MasterIngress WorkerIpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>Ingress group</td>
<td>Description</td>
<td>IP protocol</td>
<td>Port range</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>MasterIngress IpsecEsp</strong></td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td><strong>MasterIngress WorkerIpsecEsp</strong></td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td><strong>MasterIngress InternalUDP</strong></td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td><strong>MasterIngress WorkerInternalUDP</strong></td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td><strong>MasterIngress IngressServicesUDP</strong></td>
<td>Kubernetes Ingress services</td>
<td>udp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td><strong>MasterIngress WorkerIngressServicesUDP</strong></td>
<td>Kubernetes Ingress services</td>
<td>udp</td>
<td>30000 - 32767</td>
</tr>
</tbody>
</table>

**Worker Ingress**

The worker machines require the following Ingress groups. Each Ingress group is a AWS::EC2::SecurityGroupIngress resource.

<table>
<thead>
<tr>
<th>Ingress group</th>
<th>Description</th>
<th>IP protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WorkerIngress Vxlan</strong></td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td><strong>WorkerIngress WorkerVxlan</strong></td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td><strong>WorkerIngress Internal</strong></td>
<td>Internal cluster communication</td>
<td>tcp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td><strong>WorkerIngress WorkerInternal</strong></td>
<td>Internal cluster communication</td>
<td>tcp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td><strong>WorkerIngress Kube</strong></td>
<td>Kubernetes kubelet, scheduler, and controller manager</td>
<td>tcp</td>
<td>10250</td>
</tr>
<tr>
<td><strong>WorkerIngress WorkerKube</strong></td>
<td>Kubernetes kubelet, scheduler, and controller manager</td>
<td>tcp</td>
<td>10250</td>
</tr>
<tr>
<td>Ingress group</td>
<td>Description</td>
<td>IP protocol</td>
<td>Port range</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------</td>
<td>-------------</td>
<td>------------------</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>IngressServices</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>Geneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>IpsecIke</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>IpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td>IpsecEsp</td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>InternalUDP</td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>WorkerIngress</td>
<td>Kubernetes Ingress servicesUDP</td>
<td>udp</td>
<td>30000 - 32767</td>
</tr>
</tbody>
</table>
### Roles and instance profiles

You must grant the machines permissions in AWS. The provided CloudFormation templates grant the machines Allow permissions for the following **AWS::IAM::Role** objects and provide a **AWS::IAM::InstanceProfile** for each set of roles. If you do not use the templates, you can grant the machines the following broad permissions or the following individual permissions.

<table>
<thead>
<tr>
<th>Role</th>
<th>Effect</th>
<th>Action</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master</td>
<td>Allow</td>
<td>ec2:*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>elasticloadbalancing:*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>iam:PassRole</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>s3:GetObject</td>
<td>*</td>
</tr>
<tr>
<td>Worker</td>
<td>Allow</td>
<td>ec2:Describe*</td>
<td>*</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>Allow</td>
<td>ec2:Describe*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>ec2:AttachVolume</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>ec2:DetachVolume</td>
<td>*</td>
</tr>
</tbody>
</table>

#### 2.10.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

#### 2.10.3.4. Required AWS permissions

**NOTE**

Your IAM user must have the permission **tag:GetResources** in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

<table>
<thead>
<tr>
<th>Ingress group</th>
<th>Description</th>
<th>IP protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorkerIngress</td>
<td>Kubernetes Ingress services</td>
<td>udp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServicesUDP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
When you attach the **AdministratorAccess** policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

**Example 2.13. Required EC2 permissions for installation**

- `tag:TagResources`
- `tag:UntagResources`
- `ec2:AllocateAddress`
- `ec2:AssociateAddress`
- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:CopyImage`
- `ec2:CreateNetworkInterface`
- `ec2:AttachNetworkInterface`
- `ec2:CreateSecurityGroup`
- `ec2:CreateTags`
- `ec2:CreateVolume`
- `ec2:DeleteSecurityGroup`
- `ec2:DeleteSnapshot`
- `ec2:DeleteTags`
- `ec2:DeregisterImage`
- `ec2:DescribeAccountAttributes`
- `ec2:DescribeAddresses`
- `ec2:DescribeAvailabilityZones`
- `ec2:DescribeDhcpOptions`
- `ec2:DescribeImages`
- `ec2:DescribeInstanceAttribute`
- `ec2:DescribeInstanceCreditSpecifications`
- `ec2:DescribeInstances`
- `ec2:DescribeInternetGateways`
- `ec2:DescribeKeyPairs`
Example 2.14. Required permissions for creating network resources during installation

- ec2:AssociateDhcpOptions
- ec2:AssociateRouteTable
- ec2:AttachInternetGateway
- ec2:CreateDhcpOptions
- ec2:CreateInternetGateway
• ec2:CreateNatGateway
• ec2:CreateRoute
• ec2:CreateRouteTable
• ec2:CreateSubnet
• ec2:CreateVpc
• ec2:CreateVpcEndpoint
• ec2:ModifySubnetAttribute
• ec2:ModifyVpcAttribute

NOTE
If you use an existing VPC, your account does not require these permissions for creating network resources.

Example 2.15. Required Elastic Load Balancing permissions (ELB) for installation

• elasticloadbalancing:AddTags
• elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
• elasticloadbalancing:AttachLoadBalancerToSubnets
• elasticloadbalancing:ConfigureHealthCheck
• elasticloadbalancing:CreateLoadBalancer
• elasticloadbalancing:CreateLoadBalancerListeners
• elasticloadbalancing:DeleteLoadBalancer
• elasticloadbalancing:DeregisterInstancesFromLoadBalancer
• elasticloadbalancing:DescribeInstanceHealth
• elasticloadbalancing:DescribeLoadBalancerAttributes
• elasticloadbalancing:DescribeLoadBalancers
• elasticloadbalancing:DescribeTags
• elasticloadbalancing:ModifyLoadBalancerAttributes
• elasticloadbalancing:RegisterInstancesWithLoadBalancer
• elasticloadbalancing:SetLoadBalancerPoliciesOfListener

Example 2.16. Required Elastic Load Balancing permissions (ELBv2) for installation
- elasticloadbalancing:AddTags
- elasticloadbalancing:CreateListener
- elasticloadbalancing:CreateLoadBalancer
- elasticloadbalancing:CreateTargetGroup
- elasticloadbalancing:DeleteLoadBalancer
- elasticloadbalancing:DeregisterTargets
- elasticloadbalancing:DescribeListeners
- elasticloadbalancing:DescribeLoadBalancerAttributes
- elasticloadbalancing:DescribeLoadBalancers
- elasticloadbalancing:DescribeTargetGroupAttributes
- elasticloadbalancing:DescribeTargetHealth
- elasticloadbalancing:ModifyLoadBalancerAttributes
- elasticloadbalancing:ModifyTargetGroup
- elasticloadbalancing:ModifyTargetGroupAttributes
- elasticloadbalancing:RegisterTargets

Example 2.17. Required IAM permissions for installation

- iam:AddRoleToInstanceProfile
- iam:CreateInstanceProfile
- iam:CreateRole
- iam:DeleteInstanceProfile
- iam:DeleteRole
- iam:DeleteRolePolicy
- iam:GetInstanceProfile
- iam:GetRole
- iam:GetRolePolicy
- iam:GetUser
- iam:ListInstanceProfilesForRole
- iam:ListRoles
- iam:ListUsers
- iam:PassRole
- iam:PutRolePolicy
- iam:RemoveRoleFromInstanceProfile
- iam:SimulatePrincipalPolicy
- iam:TagRole

**NOTE**

If you have not created an elastic load balancer (ELB) in your AWS account, the IAM user also requires the `iam:CreateServiceLinkedRole` permission.

**Example 2.18. Required Route 53 permissions for installation**

- route53:ChangeResourceRecordSets
- route53:ChangeTagsForResource
- route53:CreateHostedZone
- route53:DeleteHostedZone
- route53:GetChange
- route53:GetHostedZone
- route53:ListHostedZones
- route53:ListHostedZonesByName
- route53:ListResourceRecordSets
- route53:ListTagsForResource
- route53:UpdateHostedZoneComment

**Example 2.19. Required S3 permissions for installation**

- s3:CreateBucket
- s3:DeleteBucket
- s3:GetAccelerateConfiguration
- s3:GetBucketAcl
- s3:GetBucketCors
- s3:GetBucketLocation
- s3:GetBucketLogging
• s3:GetBucketObjectLockConfiguration
• s3:GetBucketReplication
• s3:GetBucketRequestPayment
• s3:GetBucketTagging
• s3:GetBucketVersioning
• s3:GetBucketWebsite
• s3:GetEncryptionConfiguration
• s3:GetLifecycleConfiguration
• s3:GetReplicationConfiguration
• s3:ListBucket
• s3:PutBucketAcl
• s3:PutBucketTagging
• s3:PutEncryptionConfiguration

Example 2.20. S3 permissions that cluster Operators require

• s3:DeleteObject
• s3:GetObject
• s3:GetObjectAcl
• s3:GetObjectTagging
• s3:GetObjectVersion
• s3:PutObject
• s3:PutObjectAcl
• s3:PutObjectTagging

Example 2.21. Required permissions to delete base cluster resources

• autoscaling:DescribeAutoScalingGroups
• ec2:DeleteNetworkInterface
• ec2:DeleteVolume
• elasticloadbalancing:DeleteTargetGroup
• elasticloadbalancing:DescribeTargetGroups
Example 2.22. Required permissions to delete network resources

- ec2:DeleteDhcpOptions
- ec2:DeleteInternetGateway
- ec2:DeleteNatGateway
- ec2:DeleteRoute
- ec2:DeleteRouteTable
- ec2:DeleteSubnet
- ec2:DeleteVpc
- ec2:DeleteVpcEndpoints
- ec2:DetachInternetGateway
- ec2:DisassociateRouteTable
- ec2:ReplaceRouteTableAssociation

**NOTE**

If you use an existing VPC, your account does not require these permissions to delete network resources.

Example 2.23. Additional IAM and S3 permissions that are required to create manifests

- iam:DeleteAccessKey
- iam:DeleteUser
- iam:DeleteUserPolicy
• iam:GetUserPolicy
• iam:ListAccessKeys
• iam:PutUserPolicy
• iam:TagUser
• iam:GetUserPolicy
• iam:ListAccessKeys
• s3:PutBucketPublicAccessBlock
• s3:GetBucketPublicAccessBlock
• s3:PutLifecycleConfiguration
• s3:HeadBucket
• s3:ListBucketMultipartUploads
• s3:AbortMultipartUpload

**NOTE**

If you are managing your cloud provider credentials with mint mode, the IAM user also requires the **iam:CreateAccessKey** and **iam:CreateUser** permissions.

Example 2.24. Optional permission for quota checks for installation

• servicequotas:ListAWSDefaultServiceQuotas

### 2.10.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.
IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

2.10.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ ssh-keygen -t ed25519 -N "\" -f <path>/<file_name> 1
```

Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.
Running this command generates an SSH key that does not require a password in the location that you specified.

NOTE
If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```bash
$ eval "$(ssh-agent -s)"
```

Example output

Agent pid 31874

NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```bash
$ ssh-add <path>/<file_name>
```

Example output

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

2.10.6. Creating the installation files for AWS

To install OpenShift Container Platform on Amazon Web Services (AWS) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the install-config.yaml file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate var partition during the preparation phases of installation.

2.10.6.1. Optional: Creating a separate /var partition
It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the /var partition or a subdirectory of /var. For example:

- /var/lib/containers: Holds container-related content that can grow as more images and containers are added to a system.
- /var/lib/etcd: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- /var: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a /var directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because /var must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate /var partition by creating a machine config that is inserted during the openshift-install preparation phases of an OpenShift Container Platform installation.

**IMPORTANT**

If you follow the steps to create a separate /var partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```
   $ mkdir $HOME/clusterconfig
   ```

2. Run openshift-install to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

   ```
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ```

   **Example output**

   ```
   ? SSH Public Key ... 
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory 
   INFO Manifests created in: $HOME/clusterconfig/manifests and $HOME/clusterconfig/openshift
   ```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

   ```
   $ ls $HOME/clusterconfig/openshift/
   ```
Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  
          partitions:
            - label: var
              startMiB: <partition_start_offset>  
              sizeMiB: <partition_size>
        filesystems:
          - device: /dev/disk/by-partlabel/var
            path: /var
            format: xfs
    systemd:
      units:
        - name: var.mount  
          enabled: true
          contents:
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota
            [Install]
            WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
3. The size of the data partition in mebibytes.

4. The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-lib-containers.mount`.

5. The `prjquota` mount option must be enabled for filesystems used for container storage.

---

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

---

5. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
  auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 2.10.6.2. Creating the installation configuration file

Generate and customize the installation configuration file that the installation program needs to deploy your cluster.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program for user-provisioned infrastructure and the pull secret for your cluster.

- You checked that you are deploying your cluster to a region with an accompanying Red Hat Enterprise Linux CoreOS (RHCOS) AMI published by Red Hat. If you are deploying to a region that requires a custom AMI, such as an AWS GovCloud region, you must create the `install-config.yaml` file manually.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select **aws** as the platform to target.

iii. If you do not have an AWS profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

NOTE

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

iv. Select the AWS region to deploy the cluster to.

v. Select the base domain for the Route 53 service that you configured for your cluster.

vi. Enter a descriptive name for your cluster.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Optional: Back up the `install-config.yaml` file.

IMPORTANT

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

Additional resources

- See **Configuration and credential file settings** in the AWS documentation for more information about AWS profile and credential configuration.
2.10.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**
The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (`169.254.169.254`).

- If your cluster is on AWS, you added the `ec2.<region>.amazonaws.com`, `elasticloadbalancing.<region>.amazonaws.com`, and `s3.<region>.amazonaws.com` endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
     additionalTrustBundle:
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
   ...
   ``

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

2. A proxy URL to use for creating HTTPS connections outside the cluster.
A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

2.10.6.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
• You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ``

   **Example output**

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifets and install_dir/openshift
   ```

   1 For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines*.yaml
   ```

   By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset*.yaml
   ```

   Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

5. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

   ```yaml
   apiVersion: config.openshift.io/v1
   kind: DNS
   metadata:
     creationTimestamp: null
   name: cluster
   spec:
     baseDomain: example.openshift.com
   ```
If you do so, you must add ingress DNS records manually in a later step.

6. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   │   └── kubeconfig
│   ├── bootstrap.ign
│   ├── master.ign
│   └── metadata.json
│       └── worker.ign
```

### 2.10.7. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Amazon Web Services (AWS). The infrastructure name is also used to locate the appropriate AWS resources during an OpenShift Container Platform installation. The provided CloudFormation templates contain references to this infrastructure name, so you must extract it.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

**Procedure**

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json
```
1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
openshift-vw9j6 1
```

1. The output of this command is your cluster name and a random string.

### 2.10.8. Creating a VPC in AWS

You must create a Virtual Private Cloud (VPC) in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements, including VPN and route tables.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the VPC.

**NOTE**

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.

**Procedure**

1. Create a JSON file that contains the parameter values that the template requires:

   ```json
   [ 
   { 
     "ParameterKey": "VpcCidr", 1
     "ParameterValue": "10.0.0.0/16" 2
   }, 
   { 
     "ParameterKey": "AvailabilityZoneCount", 3
     "ParameterValue": "1" 4
   }, 
   { 
     "ParameterKey": "SubnetBits", 5
     "ParameterValue": "12" 6
   }
   ]
   ```
1. The CIDR block for the VPC.

2. Specify a CIDR block in the format `x.x.x.x/16-24`.

3. The number of availability zones to deploy the VPC in.

4. Specify an integer between 1 and 3.

5. The size of each subnet in each availability zone.

6. Specify an integer between 5 and 13, where 5 is /27 and 13 is /19.

2. Copy the template from the CloudFormation template for the VPC section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.

3. Launch the CloudFormation template to create a stack of AWS resources that represent the VPC:

   **IMPORTANT**

   You must enter the command on a single line.

   ```bash
   $ aws cloudformation create-stack --stack-name <name>  
   --template-body file://<template>.yaml  
   --parameters file://<parameters>.json
   ```

   - `<name>` is the name for the CloudFormation stack, such as `cluster-vpc`. You need the name of this stack if you remove the cluster.
   - `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
   - `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.

   **Example output**

   ```bash
   arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
   ```

4. Confirm that the template components exist:

   ```bash
   $ aws cloudformation describe-stacks --stack-name <name>
   ```

   After the `StackStatus` displays `CREATE_COMPLETE`, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<p>| VpcId       | The ID of your VPC. |</p>
<table>
<thead>
<tr>
<th><strong>PublicSubnetIds</strong></th>
<th>The IDs of the new public subnets.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PrivateSubnetIds</strong></td>
<td>The IDs of the new private subnets.</td>
</tr>
</tbody>
</table>

### 2.10.8.1. CloudFormation template for the VPC

You can use the following CloudFormation template to deploy the VPC that you need for your OpenShift Container Platform cluster.

**Example 2.25. CloudFormation template for the VPC**

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String

AvailabilityZoneCount:
  ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
  MinValue: 1
  MaxValue: 3
  Default: 1
  Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
  Type: Number

SubnetBits:
  ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
  MinValue: 5
  MaxValue: 13
  Default: 12
  Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
  Type: Number

Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
    - Label:
      default: "Network Configuration"
      Parameters:
        - VpcCidr
        - SubnetBits
      - Label:
        default: "Availability Zones"
        Parameters:
          - AvailabilityZoneCount
      ParameterLabels:
```

290
AvailabilityZoneCount:
  default: "Availability Zone Count"
VpcCidr:
  default: "VPC CIDR"
SubnetBits:
  default: "Bits Per Subnet"

Conditions:
DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
VPC:
  Type: "AWS::EC2::VPC"
  Properties:
    EnableDnsSupport: "true"
    EnableDnsHostnames: "true"
    CidrBlock: !Ref VpcCidr
PublicSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
InternetGateway:
  Type: "AWS::EC2::InternetGateway"
GatewayToInternet:
  Type: "AWS::EC2::VPCGatewayAttachment"
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref InternetGateway
PublicRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PublicRoute:
Type: "AWS::EC2::Route"
DependsOn: GatewayToInternet
Properties:
  RouteTableId: !Ref PublicRouteTable
  DestinationCidrBlock: 0.0.0.0/0
  GatewayId: !Ref InternetGateway

PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet
    RouteTableId: !Ref PublicRouteTable

PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PublicSubnet2
    RouteTableId: !Ref PublicRouteTable

PublicSubnetRouteTableAssociation3:
  Condition: DoAz3
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable

PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"

PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC

PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable

NAT:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP
        - AllocationId
    SubnetId: !Ref PublicSubnet

EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
  Route:
    Type: "AWS::EC2::Route"
Properties:
   RouteTableId:
      Ref: PrivateRouteTable
   DestinationCidrBlock: 0.0.0.0/0
   NatGatewayId:
      Ref: NAT

PrivateSubnet2:
   Type: "AWS::EC2::Subnet"
   Condition: DoAz2
   Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
         - 1
         - Fn::GetAZs: !Ref "AWS::Region"

PrivateRouteTable2:
   Type: "AWS::EC2::RouteTable"
   Condition: DoAz2
   Properties:
      VpcId: !Ref VPC

PrivateSubnetRouteTableAssociation2:
   Type: "AWS::EC2::SubnetRouteTableAssociation"
   Condition: DoAz2
   Properties:
      SubnetId: !Ref PrivateSubnet2
      RouteTableId: !Ref PrivateRouteTable2

NAT2:
   DependsOn:
      - GatewayToInternet
   Type: "AWS::EC2::NatGateway"
   Condition: DoAz2
   Properties:
      AllocationId: "Fn::GetAtt":
         - EIP2
         - AllocationId
      SubnetId: !Ref PublicSubnet2

EIP2:
   Type: "AWS::EC2::EIP"
   Condition: DoAz2
   Properties:
      Domain: vpc

Route2:
   Type: "AWS::EC2::Route"
   Condition: DoAz2
   Properties:
      RouteTableId:
         Ref: PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
         Ref: NAT2

PrivateSubnet3:
   Type: "AWS::EC2::Subnet"
   Condition: DoAz3
   Properties:
      VpcId: !Ref VPC
CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
AvailabilityZone: !Select
  - 2
  - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP3
        - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Principal: '*'
          Action:
            - '*'
          Resource:
            - '*'
        RouteTableIds:
          - !Ref PublicRouteTable
          - !Ref PrivateRouteTable
          - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
2.10.9. Creating networking and load balancing components in AWS

You must configure networking and classic or network load balancing in Amazon Web Services (AWS) that your OpenShift Container Platform cluster can use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the networking and load balancing components that your OpenShift Container Platform cluster requires. The template also creates a hosted zone and subnet tags.

You can run the template multiple times within a single Virtual Private Cloud (VPC).

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the AWS CloudFormation console.

NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.
Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

Procedure

1. Obtain the hosted zone ID for the Route 53 base domain that you specified in the `install-config.yaml` file for your cluster. You can obtain details about your hosted zone by running the following command:

   ```bash
   $ aws route53 list-hosted-zones-by-name --dns-name <route53_domain>
   ```

   For the `<route53_domain>`, specify the Route 53 base domain that you used when you generated the `install-config.yaml` file for the cluster.

   Example output

   In the example output, the hosted zone ID is `Z21IXYZABCZ2A4`.

2. Create a JSON file that contains the parameter values that the template requires:

   ```json
   [
   {
     "ParameterKey": "ClusterName",  
     "ParameterValue": "mycluster"
   },
   {
     "ParameterKey": "InfrastructureName",  
     "ParameterValue": "mycluster-<random_string>"
   },
   {
     "ParameterKey": "HostedZoneId",  
     "ParameterValue": "<random_string>"
   },
   {
     "ParameterKey": "HostedZoneName",  
     "ParameterValue": "example.com"
   },
   {
     "ParameterKey": "PublicSubnets",  
     "ParameterValue": "subnet-<random_string>"
   }
   ]
   ```

   OpenShift Container Platform 4.6 Installing
A short, representative cluster name to use for hostnames, etc.

Specify the cluster name that you used when you generated the install-config.yaml file for the cluster.

The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

The Route 53 public zone ID to register the targets with.

Specify the Route 53 public zone ID, which as a format similar to `Z21IXYZABCZ2A4`. You can obtain this value from the AWS console.

The Route 53 zone to register the targets with.

Specify the Route 53 base domain that you used when you generated the install-config.yaml file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.

The public subnets that you created for your VPC.

Specify the `PublicSubnetIds` value from the output of the CloudFormation template for the VPC.

The private subnets that you created for your VPC.

Specify the `PrivateSubnetIds` value from the output of the CloudFormation template for the VPC.

The VPC that you created for the cluster.

Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

3. Copy the template from the CloudFormation template for the network and load balancers section of this topic and save it as a YAML file on your computer. This template describes the networking and load balancing objects that your cluster requires.

**IMPORTANT**

If you are deploying your cluster to an AWS government region, you must update the `InternalApiServerRecord` in the CloudFormation template to use CNAME records. Records of type ALIAS are not supported for AWS government regions.
4. Launch the CloudFormation template to create a stack of AWS resources that provide the networking and load balancing components:

```
$ aws cloudformation create-stack --stack-name <name>  
  --template-body file://<template>.yaml  
  --parameters file://<parameters>.json  
  --capabilities CAPABILITY_NAMED_IAM
```

**IMPORTANT**

You must enter the command on a single line.

- `<name>` is the name for the CloudFormation stack, such as `cluster-dns`. You need the name of this stack if you remove the cluster.
- `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
- `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.
- You must explicitly declare the `CAPABILITY_NAMED_IAM` capability because the provided template creates some `AWS::IAM::Role` resources.

**Example output**

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the `StackStatus` displays `CREATE_COMPLETE`, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrivateHostedZoneId</td>
<td>Hosted zone ID for the private DNS.</td>
</tr>
<tr>
<td>ExternalApiLoadBalancerName</td>
<td>Full name of the external API load balancer.</td>
</tr>
<tr>
<td>InternalApiLoadBalancerName</td>
<td>Full name of the internal API load balancer.</td>
</tr>
<tr>
<td>ApiServer</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>DnsName</td>
<td></td>
</tr>
<tr>
<td>Full hostname of the API server.</td>
<td></td>
</tr>
<tr>
<td>RegisterN</td>
<td></td>
</tr>
<tr>
<td>lblpTarget</td>
<td></td>
</tr>
<tr>
<td>Lambda ARN useful to help register/deregister IP targets for these load balancers.</td>
<td></td>
</tr>
<tr>
<td>ExternalA</td>
<td></td>
</tr>
<tr>
<td>piTargetG</td>
<td></td>
</tr>
<tr>
<td>roupArn</td>
<td></td>
</tr>
<tr>
<td>ARN of external API target group.</td>
<td></td>
</tr>
<tr>
<td>InternalAp</td>
<td></td>
</tr>
<tr>
<td>iTargetGr</td>
<td></td>
</tr>
<tr>
<td>oupArn</td>
<td></td>
</tr>
<tr>
<td>ARN of internal API target group.</td>
<td></td>
</tr>
<tr>
<td>InternalSe</td>
<td></td>
</tr>
<tr>
<td>rviceTarg</td>
<td></td>
</tr>
<tr>
<td>etGroupA</td>
<td></td>
</tr>
<tr>
<td>rn</td>
<td></td>
</tr>
<tr>
<td>ARN of internal service target group.</td>
<td></td>
</tr>
</tbody>
</table>

### 2.10.9.1. CloudFormation template for the network and load balancers

You can use the following CloudFormation template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster.

#### Example 2.26. CloudFormation template for the network and load balancers

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
ClusterName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, representative cluster name to use for host names and other identifying names.
  Type: String

InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
  Type: String

HostedZoneId:
  Description: The Route53 public zone ID to register the targets with, such as
```
HostedZoneName:
  Description: The Route53 zone to register the targets with, such as example.com. Omit the trailing period.
  Type: String
  Default: "example.com"

PublicSubnets:
  Description: The internet-facing subnets.
  Type: List<AWS::EC2::Subnet::Id>

PrivateSubnets:
  Description: The internal subnets.
  Type: List<AWS::EC2::Subnet::Id>

Vpcld:
  Description: The VPC-scoped resources will belong to this VPC.
  Type: AWS::EC2::VPC::Id

Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label: "Cluster Information"
    Parameters:
      - ClusterName
      - InfrastructureName
  - Label: "Network Configuration"
    Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
  - Label: "DNS"
    Parameters:
      - HostedZoneName
      - HostedZoneId

ParameterLabels:
ClusterName:
  default: "Cluster Name"
InfrastructureName:
  default: "Infrastructure Name"
Vpcld:
  default: "VPC ID"
PublicSubnets:
  default: "Public Subnets"
PrivateSubnets:
  default: "Private Subnets"
HostedZoneName:
  default: "Public Hosted Zone Name"
HostedZoneId:
  default: "Public Hosted Zone ID"

Resources:
ExtApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
IpAddressType: ipv4
Subnets: !Ref PublicSubnets
Type: network

IntApiElb:
Type: AWS::ElasticLoadBalancingV2::LoadBalancer
Properties:
  Name: !Join ["-", [!Ref InfrastructureName, "int"]]
  Scheme: internal
  IpAddressType: ipv4
  Subnets: !Ref PrivateSubnets
  Type: network

IntDns:
Type: "AWS::Route53::HostedZone"
Properties:
  HostedZoneConfig:
    Comment: "Managed by CloudFormation"
  Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
  HostedZoneTags:
    - Key: Name
      Value: !Join ["-", [!Ref InfrastructureName, "int"]]
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
  VPCs:
    - VPCId: !Ref VpcId
  VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:
Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref HostedZoneId
  RecordSets:
    - Name: !Join [".", ["api", !Ref ClusterName, !Join [".", [!Ref HostedZoneName, "."]]]]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref IntDns
  RecordSets:
    - Name: !Join [".", ["api", !Ref ClusterName, !Join [".", [!Ref HostedZoneName, "."]]]]

Type: A
AliasTarget:
  HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
  DNSName: !GetAtt IntApiElb.DNSName
- Name:
  !Join [ 
    "",
    [
      "api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]],
    ]
Type: A
AliasTarget:
  HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
  DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
  DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
  LoadBalancerArn:
    Ref: ExtApiElb
  Port: 6443
  Protocol: TCP

ExternalApiTargetGroup:
Type: AWS::ElasticLoadBalancingV2::TargetGroup
Properties:
  HealthCheckIntervalSeconds: 10
  HealthCheckPath: "/readyz"
  HealthCheckPort: 6443
  HealthCheckProtocol: HTTPS
  HealthyThresholdCount: 2
  UnhealthyThresholdCount: 2
  Port: 6443
  Protocol: TCP
  TargetType: ip
  VpcId:
    Ref: VpcId
  TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalApiListener:
Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
  DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
  LoadBalancerArn:
    Ref: IntApiElb
  Port: 6443
  Protocol: TCP
InternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

InternalServiceInternalListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
        TargetGroupArn:
          Ref: InternalServiceTargetGroup
        LoadBalancerArn:
          Ref: IntApiElb
        Port: 22623
        Protocol: TCP

InternalServiceTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/healthz"
    HealthCheckPort: 22623
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 22623
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

RegisterTargetLambdaIamRole:
  Type: AWS::IAM::Role
  Properties:
    RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
Principal:
  Service:
    - "lambda.amazonaws.com"
Action:
  - "sts:AssumeRole"
Path: "/

Policies:
- PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]

PolicyDocument:
Version: "2012-10-17"
Statement:
- Effect: "Allow"
  Action:
  - "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
    ]
  Resource: !Ref InternalApiTargetGroup
- Effect: "Allow"
  Action:
  - "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
    ]
  Resource: !Ref InternalServiceTargetGroup
- Effect: "Allow"
  Action:
  - "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
    ]
  Resource: !Ref ExternalApiTargetGroup

RegisterNlbIpTargets:
  Type: "AWS::Lambda::Function"
Properties:
  Handler: "index.handler"
  Role:
    Fn::GetAtt:
    - "RegisterTargetLambdaIamRole"
    - "Arn"
  Code:
    ZipFile:
      import json
      import boto3
      import cfnresponse
      def handler(event, context):
        elb = boto3.client('elbv2')
        if event['RequestType'] == 'Delete':
          elb.deregister_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'], Targets=[{'Id': event['ResourceProperties']['TargetIp']},])
        elif event['RequestType'] == 'Create':
          elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'], Targets=[{'Id': event['ResourceProperties']['TargetIp']},])
        responseData = {}
        cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData, OpenShift Container Platform 4.6 Installing
event['ResourceProperties']['TargetArn'] + event['ResourceProperties']['TargetIp'])
Runtime: "python3.7"
Timeout: 120

RegisterSubnetTagsLambdaIamRole:
Type: AWS::IAM::Role
Properties:
  RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
  AssumeRolePolicyDocument:
    Version: "2012-10-17"
    Statement:
      - Effect: "Allow"
        Principal:
          Service:
            - "lambda.amazonaws.com"
        Action:
          - "sts:AssumeRole"
    Path: "/
Policies:
  - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
    PolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Action:
            ["ec2:DeleteTags", "ec2:CreateTags"]
          Resource: "arn:aws:ec2:*:*:subnet/*"
        - Effect: "Allow"
          Action:
            ["ec2:DescribeSubnets", "ec2:DescribeTags"]
          Resource: "*"

RegisterSubnetTags:
Type: "AWS::Lambda::Function"
Properties:
  Handler: "index.handler"
  Role:
    Fn::GetAtt:
      - "RegisterSubnetTagsLambdaIamRole"
      - "Arn"
  Code:
    ZipFile: |
      import json
      import boto3
      import cfnresponse
      def handler(event, context):
        ec2_client = boto3.client('ec2')
        if event['RequestType'] == 'Delete':
          for subnet_id in event['ResourceProperties']['Subnets']:
            ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +

CHAPTER 2. INSTALLING ON AWS
elif event['RequestType'] == 'Create':
    for subnet_id in event['ResourceProperties']['Subnets']:
        ec2_client.create_tags
        responseData = {}
        cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
        event['ResourceProperties']['InfrastructureName'] + event['ResourceProperties']['Subnets'][0])
        Runtime: "python3.7"
        Timeout: 120

RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
        ServiceToken: !GetAtt RegisterSubnetTags.Arn
        InfrastructureName: !Ref InfrastructureName
        Subnets: !Ref PublicSubnets

RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
        ServiceToken: !GetAtt RegisterSubnetTags.Arn
        InfrastructureName: !Ref InfrastructureName
        Subnets: !Ref PrivateSubnets

Outputs:
    PrivateHostedZoneld:
        Description: Hosted zone ID for the private DNS, which is required for private records.
        Value: !Ref IntDns
    ExternalApiLoadBalancerName:
        Description: Full name of the external API load balancer.
        Value: !GetAtt ExtApiElb.LoadBalancerFullName
    InternalApiLoadBalancerName:
        Description: Full name of the internal API load balancer.
        Value: !GetAtt IntApiElb.LoadBalancerFullName
    ApiServerDnsName:
        Description: Full hostname of the API server, which is required for the Ignition config files.
        Value: !Join ["", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]

RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn

ExternalApiTargetGroupArn:
    Description: ARN of the external API target group.
    Value: !Ref ExternalApiTargetGroup

InternalApiTargetGroupArn:
    Description: ARN of the internal API target group.
    Value: !Ref InternalApiTargetGroup

InternalServiceTargetGroupArn:
    Description: ARN of the internal service target group.
    Value: !Ref InternalServiceTargetGroup
IMPORTANT

If you are deploying your cluster to an AWS government region, you must update the `InternalApiServerRecord` to use CNAME records. Records of type ALIAS are not supported for AWS government regions. For example:

```
Type: CNAME
TTL: 10
ResourceRecords: 
- !GetAtt IntApiElb.DNSName
```

Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the AWS CloudFormation console.

- You can view details about your hosted zones by navigating to the AWS Route 53 console.

- See Listing public hosted zones in the AWS documentation for more information about listing public hosted zones.

2.10.10. Creating security group and roles in AWS

You must create security groups and roles in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the security groups and roles that your OpenShift Container Platform cluster requires.

NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.

- You added your AWS keys and region to your local AWS profile by running `aws configure`.

- You generated the Ignition config files for your cluster.

- You created and configured a VPC and associated subnets in AWS.

Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "InfrastructureName"
  }
]`
The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

The CIDR block for the VPC.

Specify the CIDR block parameter that you used for the VPC that you defined in the form `x.x.x.x/16-24`.

The private subnets that you created for your VPC.

Specify the `PrivateSubnetIds` value from the output of the CloudFormation template for the VPC.

The VPC that you created for the cluster.

Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

2. Copy the template from the CloudFormation template for security objects section of this topic and save it as a YAML file on your computer. This template describes the security groups and roles that your cluster requires.

3. Launch the CloudFormation template to create a stack of AWS resources that represent the security groups and roles:

```
$ aws cloudformation create-stack --stack-name <name>  
  --template-body file://<template>.yaml  
  --parameters file://<parameters>.json 
  --capabilities CAPABILITY_NAMED_IAM
```
<name> is the name for the CloudFormation stack, such as `cluster-sec`. You need the name of this stack if you remove the cluster.

<template> is the relative path to and name of the CloudFormation template YAML file that you saved.

<parameters> is the relative path to and name of the CloudFormation parameters JSON file.

You must explicitly declare the `CAPABILITY_NAMED_IAM` capability because the provided template creates some `AWS::IAM::Role` and `AWS::IAM::InstanceProfile` resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the `StackStatus` displays `CREATE_COMPLETE`, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>MasterSecurityGroupId</th>
<th>Master Security Group ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorkerSecurityGroupId</td>
<td>Worker Security Group ID</td>
</tr>
<tr>
<td>MasterInstanceProfile</td>
<td>Master IAM Instance Profile</td>
</tr>
<tr>
<td>WorkerInstanceProfile</td>
<td>Worker IAM Instance Profile</td>
</tr>
</tbody>
</table>

2.10.10.1. CloudFormation template for security objects

You can use the following CloudFormation template to deploy the security objects that you need for your OpenShift Container Platform cluster.

Example 2.27. CloudFormation template for security objects

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)
```
Parameters:

InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
  Type: String
  VpcCidr:
    AllowedPattern: ^((\[0-9]([-9]\[0-9]1[0-9]\[2\]0-4][0-9])2[0-4][0-9])3("16-9]\[2[0-4]])$7
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:

AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label: "Cluster Information"
  Parameters:
    - InfrastructureName
      - Label: "Network Configuration"
  Parameters:
    - VpcId
    - VpcCidr
    - PrivateSubnets
    ParameterLabels:
      InfrastructureName: "Infrastructure Name"
      VpcId: "VPC ID"
      VpcCidr: "VPC CIDR"
      PrivateSubnets: "Private Subnets"

Resources:

MasterSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Master Security Group
    SecurityGroupIngress:
    - IpProtocol: icmp
    FromPort: 0
WorkerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Worker Security Group
    SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp
Description: Vxlan packets
FromPort: 4789
ToPort: 4789
IpProtocol: udp

MasterIngressGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

MasterIngressWorkerGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

MasterIngressInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

MasterIngressWorkerInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: tcp

MasterIngressInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

MasterIngressWorkerInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Internal cluster communication
  FromPort: 9000
  ToPort: 9999
  IpProtocol: udp

MasterIngressKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Kubernetes kubelet, scheduler and controller manager
  FromPort: 10250
  ToPort: 10259
  IpProtocol: tcp

MasterIngressWorkerKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes kubelet, scheduler and controller manager
  FromPort: 10250
  ToPort: 10259
  IpProtocol: tcp

MasterIngressIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: tcp

MasterIngressWorkerIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: tcp

MasterIngressIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
ToPort: 32767
IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: udp

WorkerIngressVxlan:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Vxlan packets
  FromPort: 4789
  ToPort: 4789
  IpProtocol: udp

WorkerIngressMasterVxlan:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Vxlan packets
  FromPort: 4789
  ToPort: 4789
  IpProtocol: udp

WorkerIngressGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Geneve packets
  FromPort: 6081
  ToPort: 6081
  IpProtocol: udp

WorkerIngressMasterGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Geneve packets
  FromPort: 6081
  ToPort: 6081
  IpProtocol: udp

WorkerIngressInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
WorkerIngressMasterInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Internal cluster communication
  FromPort: 9000
  ToPort: 9999
  IpProtocol: tcp

WorkerIngressInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Internal cluster communication
  FromPort: 9000
  ToPort: 9999
  IpProtocol: udp

WorkerIngressMasterInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Internal cluster communication
  FromPort: 9000
  ToPort: 9999
  IpProtocol: udp

WorkerIngressKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes secure kubelet port
  FromPort: 10250
  ToPort: 10250
  IpProtocol: tcp

WorkerIngressWorkerKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Internal Kubernetes communication
  FromPort: 10250
  ToPort: 10250
  IpProtocol: tcp
WorkerIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressMasterIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

MasterIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "ec2.amazonaws.com"
          Action:
            - "sts:AssumeRole"
      Policies:
        - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
          PolicyDocument:
Version: "2012-10-17"
Statement:
  - Effect: "Allow"
    Action:
      - "ec2:AttachVolume"
      - "ec2:AuthorizeSecurityGroupIngress"
      - "ec2:CreateSecurityGroup"
      - "ec2:CreateTags"
      - "ec2:CreateVolume"
      - "ec2:DeleteSecurityGroup"
      - "ec2:DeleteVolume"
      - "ec2:Describe"
      - "ec2:DetachVolume"
      - "ec2:ModifyInstanceAttribute"
      - "ec2:ModifyVolume"
      - "ec2:RevokeSecurityGroupIngress"
      - "elasticloadbalancing:AddTags"
      - "elasticloadbalancing:AttachLoadBalancerToSubnets"
      - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
      - "elasticloadbalancing:CreateListener"
      - "elasticloadbalancing:CreateLoadBalancer"
      - "elasticloadbalancing:CreateLoadBalancerPolicy"
      - "elasticloadbalancing:CreateLoadBalancerListeners"
      - "elasticloadbalancing:CreateTargetGroup"
      - "elasticloadbalancing:ConfigureHealthCheck"
      - "elasticloadbalancing:DeleteListener"
      - "elasticloadbalancing:DeleteLoadBalancer"
      - "elasticloadbalancing:DeleteLoadBalancerListeners"
      - "elasticloadbalancing:DeleteTargetGroup"
      - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
      - "elasticloadbalancing:DeregisterTargets"
      - "elasticloadbalancing:Describe"
      - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
      - "elasticloadbalancing:ModifyListener"
      - "elasticloadbalancing:ModifyLoadBalancerAttributes"
      - "elasticloadbalancing:ModifyTargetGroup"
      - "elasticloadbalancing:ModifyTargetGroupAttributes"
      - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
      - "elasticloadbalancing:RegisterTargets"
      - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
      - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
      - "kms:DescribeKey"
Resource: "*"

MasterInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
      - Ref: "MasterIamRole"

WorkerIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
You can view details about the CloudFormation stacks that you create by navigating to the AWS CloudFormation console.

2.10.11. RHCOS AMIs for the AWS infrastructure

Red Hat provides Red Hat Enterprise Linux CoreOS (RHCOS) AMIs valid for the various Amazon Web Services (AWS) zones you can specify for your OpenShift Container Platform nodes.

**NOTE**

You can also install to regions that do not have a RHCOS AMI published by importing your own AMI.
### Table 2.31. RHCOS AMIs

<table>
<thead>
<tr>
<th>AWS zone</th>
<th>AWS AMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>af-south-1</td>
<td>ami-09921c9c1c36e695c</td>
</tr>
<tr>
<td>ap-east-1</td>
<td>ami-01ee8446e9af6b197</td>
</tr>
<tr>
<td>ap-northeast-1</td>
<td>ami-04e5b5722a55846ea</td>
</tr>
<tr>
<td>ap-northeast-2</td>
<td>ami-0fdc25c8a0273a742</td>
</tr>
<tr>
<td>ap-south-1</td>
<td>ami-09e3deb397cc526a8</td>
</tr>
<tr>
<td>ap-southeast-1</td>
<td>ami-0630e03f75e02eec4</td>
</tr>
<tr>
<td>ap-southeast-2</td>
<td>ami-069450613262ba03c</td>
</tr>
<tr>
<td>ca-central-1</td>
<td>ami-012518cdbd3057dfd</td>
</tr>
<tr>
<td>eu-central-1</td>
<td>ami-0bd7175ff5b1aef0c</td>
</tr>
<tr>
<td>eu-north-1</td>
<td>ami-06c9ec42d0a839ad2</td>
</tr>
<tr>
<td>eu-south-1</td>
<td>ami-0614d7440a0363d71</td>
</tr>
<tr>
<td>eu-west-1</td>
<td>ami-01b89df58b5d4d5fa</td>
</tr>
<tr>
<td>eu-west-2</td>
<td>ami-06f6e31ddd554f89d</td>
</tr>
<tr>
<td>eu-west-3</td>
<td>ami-0dc82e2517ded15a1</td>
</tr>
<tr>
<td>me-south-1</td>
<td>ami-07d181e3aa0f76067</td>
</tr>
<tr>
<td>sa-east-1</td>
<td>ami-0cd44e6dd20e6c7fa</td>
</tr>
<tr>
<td>us-east-1</td>
<td>ami-04a16d506e5b0e246</td>
</tr>
<tr>
<td>us-east-2</td>
<td>ami-0a1f868ad58ea59a7</td>
</tr>
<tr>
<td>us-west-1</td>
<td>ami-0a65d76e3a6f6622f</td>
</tr>
<tr>
<td>us-west-2</td>
<td>ami-0dd9008abadc519f1</td>
</tr>
</tbody>
</table>

### 2.10.11.1. AWS regions without a published RHCOS AMI

You can deploy an OpenShift Container Platform cluster to Amazon Web Services (AWS) regions without native support for a Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI)
or the AWS software development kit (SDK). If a published AMI is not available for an AWS region, you can upload a custom AMI prior to installing the cluster. This is required if you are deploying your cluster to an AWS government region.

If you are deploying to a non-government region that does not have a published RHCOS AMI, and you do not specify a custom AMI, the installation program copies the `us-east-1` AMI to the user account automatically. Then the installation program creates the control plane machines with encrypted EBS volumes using the default or user-specified Key Management Service (KMS) key. This allows the AMI to follow the same process workflow as published RHCOS AMIs.

A region without native support for an RHCOS AMI is not available to select from the terminal during cluster creation because it is not published. However, you can install to this region by configuring the custom AMI in the `install-config.yaml` file.

### 2.10.11.2. Uploading a custom RHCOS AMI in AWS

If you are deploying to a custom Amazon Web Services (AWS) region, you must upload a custom Red Hat Enterprise Linux CoreOS (RHCOS) Amazon Machine Image (AMI) that belongs to that region.

**Prerequisites**

- You configured an AWS account.
- You created an Amazon S3 bucket with the required IAM service role.
- You uploaded your RHCOS VMDK file to Amazon S3. The RHCOS VMDK file must be the highest version that is less than or equal to the OpenShift Container Platform version you are installing.
- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer.

**Procedure**

1. Export your AWS profile as an environment variable:

   ```bash
   $ export AWS_PROFILE=<aws_profile>  
   ```

   The AWS profile name that holds your AWS credentials, like `govcloud`.

2. Export the region to associate with your custom AMI as an environment variable:

   ```bash
   $ export AWS_DEFAULT_REGION=<aws_region>  
   ```

   The AWS region, like `us-gov-east-1`.

3. Export the version of RHCOS you uploaded to Amazon S3 as an environment variable:

   ```bash
   $ export RHCOS_VERSION=<version>  
   ```

   The RHCOS VMDK version, like `4.6.0`.
4. Export the Amazon S3 bucket name as an environment variable:

   $ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>

5. Create the `containers.json` file and define your RHCOS VMDK file:

   $ cat <<EOF > containers.json
   {
     "Description": "rhcos-$[RHCOS_VERSION]-x86_64-aws.x86_64",
     "Format": "vmdk",
     "UserBucket": {
       "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
       "S3Key": "rhcos-$[RHCOS_VERSION]-x86_64-aws.x86_64.vmdk"
     }
   }
   EOF

6. Import the RHCOS disk as an Amazon EBS snapshot:

   $ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
     --description "<description>" \
     --disk-container "file://<file_path>/containers.json"

   1. The description of your RHCOS disk being imported, like `rhcos-$[RHCOS_VERSION]-x86_64-aws.x86_64`.

   2. The file path to the JSON file describing your RHCOS disk. The JSON file should contain your Amazon S3 bucket name and key.

7. Check the status of the image import:

   $ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}

**Example output**

```json
{
  "ImportSnapshotTasks": [
    {
      "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
      "ImportTaskId": "import-snap-fh6i8uil",
      "SnapshotTaskDetail": {
        "Description": "rhcos-4.6.0-x86_64-aws.x86_64",
        "DiskImageSize": 819056640.0,
        "Format": "VMDK",
        "SnapshotId": "snap-06331325870076318",
        "Status": "completed",
        "UserBucket": {
          "S3Bucket": "external-images",
          "S3Key": "rhcos-4.6.0-x86_64-aws.x86_64.vmdk"
        }
      }
    }
  ]
}
```
Copy the **SnapshotId** to register the image.

8. Create a custom RHCOS AMI from the RHCOS snapshot:

```bash
$ aws ec2 register-image \
   --region ${AWS_DEFAULT_REGION} \ 
   --architecture x86_64 \ 
   --description "rhcos-$(RHCOS_VERSION)-x86_64-aws.x86_64" \ 
   --ena-support \ 
   --name "rhcos-$(RHCOS_VERSION)-x86_64-aws.x86_64" \ 
   --virtualization-type hvm \ 
   --root-device-name '/dev/xvda' \ 
   --block-device-mappings 'DeviceName=/dev/xvda,Ebs=\ 
   {DeleteOnTermination=true,SnapshotId=<snapshot_ID>}'
```

1. The RHCOS VMDK architecture type, like **x86_64**, **s390x**, or **ppc64le**.

2. The **Description** from the imported snapshot.

3. The name of the RHCOS AMI.

4. The **SnapshotId** from the imported snapshot.

To learn more about these APIs, see the AWS documentation for importing snapshots and creating EBS-backed AMIs.

2.10.12. Creating the bootstrap node in AWS

You must create the bootstrap node in Amazon Web Services (AWS) to use during OpenShift Container Platform cluster initialization.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the bootstrap node that your OpenShift Container Platform installation requires.

**NOTE**

If you do not use the provided CloudFormation template to create yourbootstrap node, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
• You created and configured a VPC and associated subnets in AWS.
• You created and configured DNS, load balancers, and listeners in AWS.
• You created the security groups and roles required for your cluster in AWS.

Procedure

1. Provide a location to serve the bootstrap.ign Ignition config file to your cluster. This file is located in your installation directory. One way to do this is to create an S3 bucket in your cluster’s region and upload the Ignition config file to it.

   IMPORTANT

   The provided CloudFormation Template assumes that the Ignition config files for your cluster are served from an S3 bucket. If you choose to serve the files from another location, you must modify the templates.

   IMPORTANT

   If you are deploying to a region that has endpoints that differ from the AWS SDK, or you are providing your own custom endpoints, you must use a presigned URL for your S3 bucket instead of the s3:// schema.

   NOTE

   The bootstrap Ignition config file does contain secrets, like X.509 keys. The following steps provide basic security for the S3 bucket. To provide additional security, you can enable an S3 bucket policy to allow only certain users, such as the OpenShift IAM user, to access objects that the bucket contains. You can avoid S3 entirely and serve your bootstrap Ignition config file from any address that the bootstrap machine can reach.

   a. Create the bucket:

   ```bash
   $ aws s3 mb s3://<cluster-name>-infra
   ```

   `<cluster-name>-infra` is the bucket name. When creating the install-config.yaml file, replace `<cluster-name>` with the name specified for the cluster.

   b. Upload the bootstrap.ign Ignition config file to the bucket:

   ```bash
   $ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   c. Verify that the file uploaded:

   ```bash
   $ aws s3 ls s3://<cluster-name>-infra/
   ```
Example output

2019-04-03 16:15:16  314878 bootstrap.ign

2. Create a JSON file that contains the parameter values that the template requires:

```json
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcId", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
```
1. The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

2. Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

3. Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the bootstrap node.

4. Specify a valid `AWS::EC2::Image::Id` value.

5. CIDR block to allow SSH access to the bootstrap node.

6. Specify a CIDR block in the format `x.x.x.x/16-24`.

7. The public subnet that is associated with your VPC to launch the bootstrap node into.

8. Specify the `PublicSubnetIds` value from the output of the CloudFormation template for the VPC.

9. The master security group ID (for registering temporary rules)

10. Specify the `MasterSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

11. The VPC created resources will belong to.

12. Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

13. Location to fetch bootstrap Ignition config file from.

14. Specify the S3 bucket and file name in the form `s3://<bucket_name>/bootstrap.ign`.

15. Whether or not to register a network load balancer (NLB).

16. Specify `yes` or `no`. If you specify `yes`, you must provide a Lambda Amazon Resource Name (ARN) value.

17. The ARN for NLB IP target registration lambda group.

18. Specify the `RegisterNlbIpTargetsLambda` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

19. The ARN for external API load balancer target group.

20. Specify the `ExternalApiTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.
The ARN for internal API load balancer target group.

Specify the **InternalApiTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for internal service load balancer target group.

Specify the **InternalServiceTargetGroupArn** value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

3. Copy the template from the **CloudFormation template for the bootstrap machines** section of this topic and save it as a YAML file on your computer. This template describes the bootstrap machine that your cluster requires.

4. Launch the CloudFormation template to create a stack of AWS resources that represent the bootstrap node:

```bash
$ aws cloudformation create-stack --stack-name <name>  
   --template-body file://<template>.yaml  
   --parameters file://<parameters>.json  
   --capabilities CAPABILITY_NAMED_IAM
```

1. **<name>** is the name for the CloudFormation stack, such as `cluster-bootstrap`. You need the name of this stack if you remove the cluster.

2. **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.

3. **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

4. You must explicitly declare the **CAPABILITY_NAMED_IAM** capability because the provided template creates some **AWS::IAM::Role** and **AWS::IAM::InstanceProfile** resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

OpenShift Container Platform 4.6 Installing
After the StackStatus displays CREATE_COMPLETE, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap InstanceId</td>
<td>The bootstrap Instance ID.</td>
</tr>
<tr>
<td>Bootstrap PublicIp</td>
<td>The bootstrap node public IP address.</td>
</tr>
<tr>
<td>Bootstrap PrivateIp</td>
<td>The bootstrap node private IP address.</td>
</tr>
</tbody>
</table>

2.10.12.1. CloudFormation template for the bootstrap machine

You can use the following CloudFormation template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster.

Example 2.28. CloudFormation template for the bootstrap machine

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$  
  MaxLength: 27  
  MinLength: 1  
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.  
  Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.  
  Type: String
RhcosAmi:
  Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.  
  Type: AWS::EC2::Image::Id
AllowedBootstrapSshCidr:
  AllowedPattern: ^((0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.(\d{0,2}|(25[0-5]|1[0-9][0-9]|1[0-9][0-9]|0[0-9][0-9] )|3)\(0-9\]\[1-9]\[0-9\]\[0-9\]\[2\]|2\[0-4\]\[0-9\]\[25\][0-5\]|(\(0-9\]\[1-9\]|2\[0-9\]|3\[0-2\]))$  
  ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.  
  Default: 0.0.0.0/0
Description: CIDR block to allow SSH access to the bootstrap node.  
  Type: String
PublicSubnet:
  Description: The public subnet to launch the bootstrap node into.  
  Type: AWS::EC2::Subnet::Id
MasterSecurityGroupId:
  Description: The master security group ID for registering temporary rules.  
  Type: AWS::EC2::SecurityGroup::Id
VpcId:
  Description: The VPC-scoped resources will belong to this VPC.  
  Type: AWS::EC2::VPC::Id
BootstrapIgnitionLocation:
  Default: s3://my-s3-bucket/bootstrap.ign
```
Description: Ignition config file location.
Type: String
AutoRegisterELB:
Default: "yes"
AllowedValues:
- "yes"
- "no"
Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
Type: String
RegisterNlbIpTargetsLambdaArn:
Description: ARN for NLB IP target registration lambda.
Type: String
ExternalApiTargetGroupArn:
Description: ARN for external API load balancer target group.
Type: String
InternalApiTargetGroupArn:
Description: ARN for internal API load balancer target group.
Type: String
InternalServiceTargetGroupArn:
Description: ARN for internal service load balancer target group.
Type: String

Metadata:
AWS::CloudFormation::Interface:
ParameterGroups:
- Label: default: "Cluster Information"
Parameters:
- InfrastructureName
- Label: default: "Host Information"
Parameters:
- RhcosAmi
- BootstrapIgnitionLocation
- MasterSecurityGroupId
- Label: default: "Network Configuration"
Parameters:
- VpcId
- AllowedBootstrapSshCidr
- PublicSubnet
- Label: default: "Load Balancer Automation"
Parameters:
- AutoRegisterELB
- RegisterNlbIpTargetsLambdaArn
- ExternalApiTargetGroupArn
- InternalApiTargetGroupArn
- InternalServiceTargetGroupArn
ParameterLabels:
InfrastructureName: default: "Infrastructure Name"
VpcId: default: "VPC ID"
AllowedBootstrapSshCidr: default: "Allowed SSH Source"
PublicSubnet:
  default: "Public Subnet"

RhcosAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"

BootstrapIgnitionLocation:
  default: "Bootstrap Ignition Source"

MasterSecurityGroupId:
  default: "Master Security Group ID"

AutoRegisterELB:
  default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:

BootstrapIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "ec2.amazonaws.com"
          Action:
            - "sts:AssumeRole"
          Path: "/"  
    Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
            - Effect: "Allow"
              Action: "ec2:Describe*"
              Resource: "*"
            - Effect: "Allow"
              Action: "ec2:AttachVolume"
              Resource: "*"
            - Effect: "Allow"
              Action: "ec2:DetachVolume"
              Resource: "*"
            - Effect: "Allow"
              Action: "s3:GetObject"
              Resource: "*"

BootstrapInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Path: "/"
    Roles:
      - Ref: "BootstrapIamRole"

BootstrapSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
GroupDescription: Cluster Bootstrap Security Group
SecurityGroupIngress:
  - IpProtocol: tcp
    FromPort: 22
    ToPort: 22
    CidrIp: !Ref AllowedBootstrapSshCidr
  - IpProtocol: tcp
    FromPort: 19531
    ToPort: 19531
    CidrIp: 0.0.0.0/0
VpcId: !Ref VpcId

BootstrapInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    IamInstanceProfile: !Ref BootstrapInstanceProfile
    InstanceType: "i3.large"
  NetworkInterfaces:
    - AssociatePublicIpAddress: "true"
      DeviceIndex: "0"
      GroupSet:
        - !Ref "BootstrapSecurityGroupId"
        - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "PublicSubnet"
  UserData:
    Fn::Base64: !Sub
      - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}}},"version":"3.1.0"}'
      - { S3Loc: !Ref BootstrapIgnitionLocation }

RegisterBootstrapApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstancePrivateIp

RegisterBootstrapInternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt BootstrapInstancePrivateIp
2.10.13. Creating the control plane machines in AWS

You must create the control plane machines in Amazon Web Services (AWS) that your cluster will use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the control plane nodes.

**IMPORTANT**

The CloudFormation template creates a stack that represents three control plane nodes.

**NOTE**

If you do not use the provided CloudFormation template to create your control plane nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
Procedure

1. Create a JSON file that contains the parameter values that the template requires:

```json
[
  {
    "ParameterKey": "InfrastructureName",  1
    "ParameterValue": "mycluster-<random_string>"  2
  },
  {
    "ParameterKey": "RhcosAmi",  3
    "ParameterValue": "ami-<random_string>"  4
  },
  {
    "ParameterKey": "AutoRegisterDNS",  5
    "ParameterValue": "yes"  6
  },
  {
    "ParameterKey": "PrivateHostedZoneId",  7
    "ParameterValue": "<random_string>"  8
  },
  {
    "ParameterKey": "PrivateHostedZoneName",  9
    "ParameterValue": "mycluster.example.com"  10
  },
  {
    "ParameterKey": "Master0Subnet",  11
    "ParameterValue": "subnet-<random_string>"  12
  },
  {
    "ParameterKey": "Master1Subnet",  13
    "ParameterValue": "subnet-<random_string>"  14
  },
  {
    "ParameterKey": "Master2Subnet",  15
    "ParameterValue": "subnet-<random_string>"  16
  },
  {
    "ParameterKey": "MasterSecurityGroupId",  17
    "ParameterValue": "sg-<random_string>"  18
  },
  {
    "ParameterKey": "IgnitionLocation",  19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"  20
  },
  {
    "ParameterKey": "CertificateAuthorities",  21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz="  22
  },
  {
    "ParameterKey": "MasterInstanceProfileName",  23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>"  24
  }
]
```
The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>.

Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the control plane machines.

Specify an AWS::EC2::Image::Id value.

Whether or not to perform DNS etcd registration.

Specify yes or no. If you specify yes, you must provide hosted zone information.

The Route 53 private zone ID to register the etcd targets with.

Specify the PrivateHostedZoneId value from the output of the CloudFormation template for DNS and load balancing.

The Route 53 zone to register the targets with.

Specify `<cluster_name>-<domain_name>` where `<domain_name>` is the Route 53 base domain that you used when you generated `install-config.yaml` file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
A subnet, preferably private, to launch the control plane machines on.

Specify a subnet from the `PrivateSubnets` value from the output of the CloudFormation template for DNS and load balancing.

The master security group ID to associate with control plane nodes (also known as the master nodes).

Specify the `MasterSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

The location to fetch control plane Ignition config file from.

Specify the generated Ignition config file location, `https://api-int.<cluster_name>.<domain_name>:22623/config/master`.

The base64 encoded certificate authority string to use.

Specify the value from the `master.ign` file that is in the installation directory. This value is the long string with the format `data:text/plain;charset=utf-8;base64,ABC…xYz==`.

The IAM profile to associate with control plane nodes.

Specify the `MasterInstanceProfile` parameter value from the output of the CloudFormation template for the security group and roles.

The type of AWS instance to use for the control plane machines.

Allowed values:

- `m4.xlarge`
- `m4.2xlarge`
- `m4.4xlarge`
- `m4.8xlarge`
- `m4.10xlarge`
- `m4.16xlarge`
- `m5.xlarge`
- `m5.2xlarge`
- `m5.4xlarge`
- `m5.8xlarge`
- `m5.10xlarge`
- `m5.16xlarge`
- `m6i.xlarge`
- `c4.2xlarge`
• c4.4xlarge
• c4.8xlarge
• r4.xlarge
• r4.2xlarge
• r4.4xlarge
• r4.8xlarge
• r4.16xlarge

**IMPORTANT**

If m4 instance types are not available in your region, such as with eu-west-3, specify an m5 type, such as m5.xlarge, instead.

Whether or not to register a network load balancer (NLB).

Specify yes or no. If you specify yes, you must provide a Lambda Amazon Resource Name (ARN) value.

The ARN for NLB IP target registration lambda group.

Specify the `RegisterNlbIpTargetsLambda` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for external API load balancer target group.

Specify the `ExternalApiTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for internal API load balancer target group.

Specify the `InternalApiTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for internal service load balancer target group.

Specify the `InternalServiceTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

2. Copy the template from the CloudFormation template for control plane machines section of this topic and save it as a YAML file on your computer. This template describes the control plane machines that your cluster requires.

3. If you specified an m5 instance type as the value for `MasterInstanceType`, add that instance type to the `MasterInstanceType.AllowedValues` parameter in the CloudFormation template.
4. Launch the CloudFormation template to create a stack of AWS resources that represent the control plane nodes:

**IMPORTANT**

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
   --template-body file://<template>.yaml 2
   --parameters file://<parameters>.json 3
```

1. `<name>` is the name for the CloudFormation stack, such as `cluster-control-plane`. You need the name of this stack if you remove the cluster.

2. `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.

3. `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.

**Example output**

```none
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```

**NOTE**

The CloudFormation template creates a stack that represents three control plane nodes.

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

### 2.10.13.1. CloudFormation template for control plane machines

You can use the following CloudFormation template to deploy the control plane machines that you need for your OpenShift Container Platform cluster.

**Example 2.29. CloudFormation template for control plane machines**

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
```
Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
Type: String

RhcosAmi:
Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
Type: AWS::EC2::Image::Id

AutoRegisterDNS:
Default: "yes"
AllowedValues:
- "yes"
- "no"
Description: Do you want to invoke DNS etcd registration, which requires Hosted Zone information?
Type: String

PrivateHostedZoneId:
Description: The Route53 private zone ID to register the etcd targets with, such as Z21IXYZABCZ2A4.
Type: String

PrivateHostedZoneName:
Description: The Route53 zone to register the targets with, such as cluster.example.com. Omit the trailing period.
Type: String

Master0Subnet:
Description: The subnets, recommend private, to launch the master nodes into.
Type: AWS::EC2::Subnet::Id

Master1Subnet:
Description: The subnets, recommend private, to launch the master nodes into.
Type: AWS::EC2::Subnet::Id

Master2Subnet:
Description: The subnets, recommend private, to launch the master nodes into.
Type: AWS::EC2::Subnet::Id

MasterSecurityGroupId:
Description: The master security group ID to associate with master nodes.
Type: AWS::EC2::SecurityGroup::Id

IgnitionLocation:
Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
Description: Ignition config file location.
Type: String

CertificateAuthorities:
Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
Description: Base64 encoded certificate authority string to use.
Type: String

MasterInstanceProfileName:
Description: IAM profile to associate with master nodes.
Type: String

MasterInstanceType:
Default: m5.xlarge
Type: String

AllowedValues:
- "m4.xlarge"
- "m4.2xlarge"
- "m4.4xlarge"
- "m4.10xlarge"
- "m4.16xlarge"
- "m5.xlarge"
- "m5.2xlarge"
- "m5.4xlarge"
- "m5.8xlarge"
- "m5.12xlarge"
- "m5.16xlarge"
- "m5a.xlarge"
- "m5a.2xlarge"
- "m5a.4xlarge"
- "m5a.8xlarge"
- "m5a.10xlarge"
- "m5a.16xlarge"
- "c4.2xlarge"
- "c4.4xlarge"
- "c4.8xlarge"
- "c5.2xlarge"
- "c5.4xlarge"
- "c5.9xlarge"
- "c5.12xlarge"
- "c5.18xlarge"
- "c5.24xlarge"
- "c5a.2xlarge"
- "c5a.4xlarge"
- "c5a.8xlarge"
- "c5a.12xlarge"
- "c5a.16xlarge"
- "c5a.24xlarge"
- "r4.xlarge"
- "r4.2xlarge"
- "r4.4xlarge"
- "r4.8xlarge"
- "r4.16xlarge"
- "r5.xlarge"
- "r5.2xlarge"
- "r5.4xlarge"
- "r5.8xlarge"
- "r5.12xlarge"
- "r5.16xlarge"
- "r5.24xlarge"
- "r5a.xlarge"
- "r5a.2xlarge"
- "r5a.4xlarge"
- "r5a.8xlarge"
- "r5a.12xlarge"
- "r5a.16xlarge"
- "r5a.24xlarge"

AutoRegisterELB:
  Default: "yes"
  AllowedValues:
  - "yes"
  - "no"
  Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
  Type: String

RegisterNlbIpTargetsLambdaArn:
  Description: ARN for NLB IP target registration lambda. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
  Type: String

ExternalApiTargetGroupArn:
Description: ARN for external API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
Type: String

InternalApiTargetGroupArn:
Description: ARN for internal API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
Type: String

InternalServiceTargetGroupArn:
Description: ARN for internal service load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
Type: String

Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label: default: "Cluster Information"
    Parameters:
    - InfrastructureName
  - Label: default: "Host Information"
    Parameters:
    - MasterInstanceType
    - RhcosAmi
    - IgnitionLocation
    - CertificateAuthorities
    - MasterSecurityGroupId
    - MasterInstanceProfileName
  - Label: default: "Network Configuration"
    Parameters:
    - VpcId
    - AllowedBootstrapSshCidr
    - Master0Subnet
    - Master1Subnet
    - Master2Subnet
  - Label: default: "DNS"
    Parameters:
    - AutoRegisterDNS
    - PrivateHostedZoneName
    - PrivateHostedZoneId
  - Label: default: "Load Balancer Automation"
    Parameters:
    - AutoRegisterELB
    - RegisterNlbIpTargetsLambdaArn
    - ExternalApiTargetGroupArn
    - InternalApiTargetGroupArn
    - InternalServiceTargetGroupArn

ParameterLabels:
InfrastructureName: default: "Infrastructure Name"
VpcId: default: "VPC ID"
Master0Subnet:
default: "Master-0 Subnet"
Master1Subnet:
  default: "Master-1 Subnet"
Master2Subnet:
  default: "Master-2 Subnet"
MasterInstanceType:
  default: "Master Instance Type"
MasterInstanceProfileName:
  default: "Master Instance Profile Name"
RhcosAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"
BootstrapIgnitionLocation:
  default: "Master Ignition Source"
CertificateAuthorities:
  default: "Ignition CA String"
MasterSecurityGroupIdd:
  default: "Master Security Group ID"
AutoRegisterDNS:
  default: "Use Provided DNS Automation"
AutoRegisterELB:
  default: "Use Provided ELB Automation"
PrivateHostedZoneName:
  default: "Private Hosted Zone Name"
PrivateHostedZoneld:
  default: "Private Hosted Zone ID"

Conditions:
DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
DoDns: !Equals ["yes", !Ref AutoRegisterDNS]

Resources:
Master0:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
    - AssociatePublicIpAddress: "false"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "MasterSecurityGroupIdd"
      SubnetId: !Ref "Master0Subnet"
    UserData:
      Fn::Base64: !Sub
        - "{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
          {"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}

SOURCE: !Ref IgnitionLocation,
CA_BUNDLE: !Ref CertificateAuthorities,
Tags:
- Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
  Value: "shared"

RegisterMaster0:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

Master1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master1Subnet"
    UserData:
      Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls": "\n          {"certificateAuthorities": ["${CA_BUNDLE}"],"version":"3.1.0"}}}
            - {"CA_BUNDLE": !Ref CertificateAuthorities}
        }'
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
  }
  Tags:
  - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
    Value: "shared"
RegisterMaster1:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalApiTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalServiceTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalServiceTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

Master2:
Type: AWS::EC2::Instance
Properties:
  ImageId: !Ref RhcosAmi
  BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
    UserData:
      Fn::Base64: !Sub
        - "{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}
          - {
              SOURCE: !Ref IgnitionLocation,
              CA_BUNDLE: !Ref CertificateAuthorities,
            }
        }
    Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

RegisterMaster2:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master2.PrivateIp

EtcdSrvRecords:
  Condition: DoDns
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneId: !Ref PrivateHostedZoneId
    Name: !Join [".", ["_etcd-server-ssl._tcp", !Ref PrivateHostedZoneName]]
    ResourceRecords:
    - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]],
    ]
    - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]],
    ]
    - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]],
    ]
    TTL: 60
    Type: SRV

Etcd0Record:
  Condition: DoDns
  Type: AWS::Route53::RecordSet
  Properties:
    HostedZoneId: !Ref PrivateHostedZoneId
    Name: !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]
    ResourceRecords:
    - !GetAtt Master0.PrivateIp
    TTL: 60
    Type: A

Etcd1Record:
You can view details about the CloudFormation stacks that you create by navigating to the AWS CloudFormation console.

2.10.14. Creating the worker nodes in AWS

You can create worker nodes in Amazon Web Services (AWS) for your cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent a worker node.

**IMPORTANT**

The CloudFormation template creates a stack that represents one worker node. You must create a stack for each worker node.

**NOTE**

If you do not use the provided CloudFormation template to create your worker nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.
Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.

Procedure

1. Create a JSON file that contains the parameter values that the CloudFormation template requires:

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker" 10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
```

```
The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the worker nodes.

Specify an `AWS::EC2::Image::Id` value.

A subnet, preferably private, to launch the worker nodes on.

Specify a subnet from the `PrivateSubnets` value from the output of the CloudFormation template for DNS and load balancing.

The worker security group ID to associate with worker nodes.

Specify the `WorkerSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

The location to fetch bootstrap Ignition config file from.

Specify the generated Ignition config location, `https://api-int.<cluster_name>.<domain_name>:22623/config/worker`.

Base64 encoded certificate authority string to use.

Specify the value from the `worker.ign` file that is in the installation directory. This value is the long string with the format `data:text/plain;charset=utf-8;base64,ABC...Yz==`.

The IAM profile to associate with worker nodes.

Specify the `WorkerInstanceProfile` parameter value from the output of the CloudFormation template for the security group and roles.

The type of AWS instance to use for the control plane machines.

Allowed values:

- `m4.large`
- `m4.xlarge`
- `m4.2xlarge`
- `m4.4xlarge`
- `m4.8xlarge`
- `m4.10xlarge`
- `m4.16xlarge`
- m5.large
- m5.xlarge
- m5.2xlarge
- m5.4xlarge
- m5.8xlarge
- m5.10xlarge
- m5.16xlarge
- m6i.xlarge
- c4.2xlarge
- c4.4xlarge
- c4.8xlarge
- r4.large
- r4.xlarge
- r4.2xlarge
- r4.4xlarge
- r4.8xlarge
- r4.16xlarge

**IMPORTANT**

If m4 instance types are not available in your region, such as with eu-west-3, use m5 types instead.

2. Copy the template from the CloudFormation template for worker machines section of this topic and save it as a YAML file on your computer. This template describes the networking objects and load balancers that your cluster requires.

3. If you specified an m5 instance type as the value for WorkerInstanceType, add that instance type to the WorkerInstanceType.AllowedValues parameter in the CloudFormation template.

4. Launch the CloudFormation template to create a stack of AWS resources that represent a worker node:

**IMPORTANT**

You must enter the command on a single line.
$ aws cloudformation create-stack --stack-name <name>  
  --template-body file://<template>.yaml  
  --parameters file://<parameters>.json

1. **<name>** is the name for the CloudFormation stack, such as `cluster-worker-1`. You need the name of this stack if you remove the cluster.

2. **<template>** is the relative path to and name of the CloudFormation template YAML file that you saved.

3. **<parameters>** is the relative path to and name of the CloudFormation parameters JSON file.

**Example output**

```
```

**NOTE**

The CloudFormation template creates a stack that represents one worker node.

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6. Continue to create worker stacks until you have created enough worker machines for your cluster. You can create additional worker stacks by referencing the same template and parameter files and specifying a different stack name.

**IMPORTANT**

You must create at least two worker machines, so you must create at least two stacks that use this CloudFormation template.

### 2.10.14.1. CloudFormation template for worker machines

You can use the following CloudFormation template to deploy the worker machines that you need for your OpenShift Container Platform cluster.

**Example 2.30. CloudFormation template for worker machines**

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$  
  MaxLength: 27  
  MinLength: 1  
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
```
Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
Type: String

RhcosAmi:
Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
Type: AWS::EC2::Image::Id

Subnet:
Description: The subnets, recommend private, to launch the master nodes into.
Type: AWS::EC2::Subnet::Id

WorkerSecurityGroupId:
Description: The master security group ID to associate with master nodes.
Type: AWS::EC2::SecurityGroup::Id

IgnitionLocation:
Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
Description: Ignition config file location.
Type: String

CertificateAuthorities:
Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
Description: Base64 encoded certificate authority string to use.
Type: String

WorkerInstanceProfileName:
Description: IAM profile to associate with master nodes.
Type: String

WorkerInstanceType:
Default: m5.large
Type: String
AllowedValues:
- "m4.large"
- "m4.xlarge"
- "m4.2xlarge"
- "m4.4xlarge"
- "m4.10xlarge"
- "m4.16xlarge"
- "m5.large"
- "m5.xlarge"
- "m5.2xlarge"
- "m5.4xlarge"
- "m5.8xlarge"
- "m5.12xlarge"
- "m5.16xlarge"
- "m5a.large"
- "m5a.xlarge"
- "m5a.2xlarge"
- "m5a.4xlarge"
- "m5a.8xlarge"
- "m5a.10xlarge"
- "m5a.16xlarge"
- "c4.large"
- "c4.xlarge"
- "c4.2xlarge"
- "c4.4xlarge"
- "c4.8xlarge"
- "c5.large"
- "c5.xlarge"
- "c5.2xlarge"
- "c5.4xlarge"
- "c5.9xlarge"
- "c5.12xlarge"
- "c5.18xlarge"
- "c5.24xlarge"
- "c5a.large"
- "c5a.xlarge"
- "c5a.2xlarge"
- "c5a.4xlarge"
- "c5a.8xlarge"
- "c5a.12xlarge"
- "c5a.16xlarge"
- "c5a.24xlarge"
- "r4.large"
- "r4.xlarge"
- "r4.2xlarge"
- "r4.4xlarge"
- "r4.8xlarge"
- "r4.16xlarge"
- "r5.large"
- "r5.xlarge"
- "r5.2xlarge"
- "r5.4xlarge"
- "r5.8xlarge"
- "r5.12xlarge"
- "r5.16xlarge"
- "r5.24xlarge"
- "r5a.large"
- "r5a.xlarge"
- "r5a.2xlarge"
- "r5a.4xlarge"
- "r5a.8xlarge"
- "r5a.12xlarge"
- "r5a.16xlarge"
- "r5a.24xlarge"
- "t3.large"
- "t3.xlarge"
- "t3.2xlarge"
- "t3a.large"
- "t3a.xlarge"
- "t3a.2xlarge"

Metadata:
AWS::CloudFormation::Interface:
ParameterGroups:
- Label: "default: "Cluster Information"
Parameters:
- InfrastructureName
- Label: "default: "Host Information"
Parameters:
- WorkerInstanceType
- RhcosAmi
- IgnitionLocation
- CertificateAuthorities
- WorkerSecurityGroupld
- WorkerInstanceProfileName
- Label:
  default: "Network Configuration"
Parameters:
- Subnet
ParameterLabels:
  Subnet:
    default: "Subnet"
  InfrastructureName:
    default: "Infrastructure Name"
  WorkerInstanceType:
    default: "Worker Instance Type"
  WorkerInstanceProfileName:
    default: "Worker Instance Profile Name"
  RhcosAmi:
    default: "Red Hat Enterprise Linux CoreOS AMI ID"
  IgnitionLocation:
    default: "Worker Ignition Source"
  CertificateAuthorities:
    default: "Ignition CA String"
  WorkerSecurityGroupId:
    default: "Worker Security Group ID"

Resources:
Worker0:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
        IamInstanceProfile: !Ref WorkerInstanceProfileName
        InstanceType: !Ref WorkerInstanceType
        NetworkInterfaces:
          - AssociatePublicIpAddress: "false"
            DeviceIndex: "0"
            GroupSet:
              - !Ref "WorkerSecurityGroupId"
            SubnetId: !Ref "Subnet"
        UserData:
          Fn::Base64: !Sub
            "\n:Base64: !Sub
            - '{"ignition":{"config":{"merge":["source":"${SOURCE}"],"security":{"tls":
            {"certificateAuthorities":["source":"${CA_BUNDLE}"],"version":"3.1.0"}}}},"version":"3.1.0"}'}
            - {"source":"${SOURCE}"},
            CA_BUNDLE: !Ref CertificateAuthorities,
        Tags:
          - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
            Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
Additional resources

- You can view details about the CloudFormation stacks that you create by navigating to the AWS CloudFormation console.

2.10.15. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure

After you create all of the required infrastructure in Amazon Web Services (AWS), you can start the bootstrap sequence that initializes the OpenShift Container Platform control plane.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.
- You created the control plane machines.
- You created the worker nodes.

Procedure

1. Change to the directory that contains the installation program and start the bootstrap process that initializes the OpenShift Container Platform control plane:

   ```
   $ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
   --log-level=info  2
   ```

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

```
INFO Waiting up to 20m0s for the Kubernetes API at https://api.mycluster.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```
If the command exits without a **FATAL** warning, your OpenShift Container Platform control plane has initialized.

**NOTE**

After the control plane initializes, it sets up the compute nodes and installs additional services in the form of Operators.

**Additional resources**

- See [Monitoring installation progress](#) for details about monitoring the installation, bootstrap, and control plane logs as an OpenShift Container Platform installation progresses.
- See [Gathering bootstrap node diagnostic data](#) for information about troubleshooting issues related to the bootstrap process.
- You can view details about the running instances that are created by using the [AWS EC2 console](#).

### 2.10.16. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) in order to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of **oc**.

#### 2.10.16.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Linux Client** entry and save the file.
4. Unpack the archive:
   
   ```
   $ tar xvf <file>
   ```
5. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the **oc** command:
2.10.16.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the oc binary to a directory that is on your PATH.

To check your PATH, open the command prompt and execute the following command:

\$ path

After you install the OpenShift CLI, it is available using the oc command:

\$ oc <command>

2.10.16.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.

To check your PATH, open a terminal and execute the following command:

\$ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

\$ oc <command>

2.10.17. Logging in to the cluster by using the CLI
You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ``

   **Example output**

   `system:admin`

2.10.18. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  63m  v1.19.0
   master-1  Ready     master  63m  v1.19.0
   master-2  Ready     master  64m  v1.19.0
   ```

   The output lists all of the machines that you created.
NOTE
The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

NOTE
Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

NOTE
For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>
```
<csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>  
```

<csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>
NOTE

It can take a few minutes after approval of the server CSRs for the machines to transition to the Ready status.

Additional information

- For more information on CSRs, see Certificate Signing Requests.

2.10.19. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

   $ watch -n5 oc get clusteroperators

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>
2. Configure the Operators that are not available.

2.10.19.1. Image registry storage configuration

Amazon Web Services provides default storage, which means the Image Registry Operator is available after installation. However, if the Registry Operator cannot create an S3 bucket and automatically configure storage, you must manually configure registry storage.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

You can configure registry storage for user-provisioned infrastructure in AWS to deploy OpenShift Container Platform to hidden regions. See **Configuring the registry for AWS user-provisioned infrastructure** for more information.

2.10.19.1.1. Configuring registry storage for AWS with user-provisioned infrastructure

During installation, your cloud credentials are sufficient to create an Amazon S3 bucket and the Registry Operator will automatically configure storage.

If the Registry Operator cannot create an S3 bucket and automatically configure storage, you can create an S3 bucket and configure storage with the following procedure.

**Prerequisites**

- You have a cluster on AWS with user-provisioned infrastructure.
- For Amazon S3 storage, the secret is expected to contain two keys:
  - **REGISTRY_STORAGE_S3_ACCESSKEY**
  - **REGISTRY_STORAGE_S3_SECRETKEY**

**Procedure**

Use the following procedure if the Registry Operator cannot create an S3 bucket and automatically configure storage.

1. Set up a **Bucket Lifecycle Policy** to abort incomplete multipart uploads that are one day old.

2. Fill in the storage configuration in `configs.imageregistry.operator.openshift.io/cluster`:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io/cluster
   ```
Example configuration

```yaml
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

**WARNING**
To secure your registry images in AWS, block public access to the S3 bucket.

2.10.19.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

  ```bash
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage":{"emptyDir":[]}}}'
  ```

**WARNING**
Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

2.10.20. Deleting the bootstrap resources

After you complete the initial Operator configuration for the cluster, remove the bootstrap resources from Amazon Web Services (AWS).

**Prerequisites**

- You completed the initial Operator configuration for your cluster.

**Procedure**
Procedure

1. Delete the bootstrap resources. If you used the CloudFormation template, delete its stack:

   - Delete the stack by using the AWS CLI:

     $ aws cloudformation delete-stack --stack-name <name> ①

     ① <name> is the name of your bootstrap stack.

   - Delete the stack by using the AWS CloudFormation console.

2.10.21. Creating the Ingress DNS Records

If you removed the DNS Zone configuration, manually create DNS records that point to the Ingress load balancer. You can create either a wildcard record or specific records. While the following procedure uses A records, you can use other record types that you require, such as CNAME or alias.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Amazon Web Services (AWS) that uses infrastructure that you provisioned.

- You installed the OpenShift CLI (oc).

- You installed the jq package.

- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix).

Procedure

1. Determine the routes to create.

   - To create a wildcard record, use *.apps.<cluster_name>.<domain_name>, where <cluster_name> is your cluster name, and <domain_name> is the Route 53 base domain for your OpenShift Container Platform cluster.

   - To create specific records, you must create a record for each route that your cluster uses, as shown in the output of the following command:

     $ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}

     {"\n"\n}{end}{end}' routes

Example output

oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
2. Retrieve the Ingress Operator load balancer status and note the value of the external IP address that it uses, which is shown in the `EXTERNAL-IP` column:

   $ oc -n openshift-ingress get service router-default

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>router-default</td>
<td>LoadBalancer</td>
<td>172.30.62.215</td>
<td>ab3...28.us-east-2.elb.amazonaws.com</td>
<td>80:31499/TCP,443:30693/TCP</td>
</tr>
</tbody>
</table>

3. Locate the hosted zone ID for the load balancer:

   $ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName == "<external_ip>").CanonicalHostedZoneNameID'

   **Example output**

   Z3AADJGX6KTTL2

   The output of this command is the load balancer hosted zone ID.

4. Obtain the public hosted zone ID for your cluster's domain:

   $ aws route53 list-hosted-zones-by-name \  
   --dns-name "<domain_name>" \  
   --query 'HostedZones[? Config.PrivateZone != `true` && Name == `<domain_name>`.].Id' \  
   --output text

   **Example output**

   /hostedzone/Z3URY6TWQ91KV

   The public hosted zone ID for your domain is shown in the command output. In this example, it is Z3URY6TWQ91KV.

5. Add the alias records to your private zone:

   $ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --change-batch '{  
   "Changes": [  
   {  
   "Action": "CREATE",

---

362
For `<private_hosted_zone_id>`, specify the value from the output of the CloudFormation template for DNS and load balancing.

For `<cluster_domain>`, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.

For `<hosted_zone_id>`, specify the public hosted zone ID for the load balancer that you obtained.

For `<external_ip>`, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6. Add the records to your public zone:

```bash
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --change-batch '{
  "Changes": [
    {
      "Action": "CREATE",
      "ResourceRecordSet": {
        "Name": "\052.apps.<cluster_domain>",
        "Type": "A",
        "AliasTarget": {
          "HostedZoneId": "<hosted_zone_id>",
          "DNSName": "<external_ip>",
          "EvaluateTargetHealth": false
        }
      }
    }
  ]
}'
```

For `<public_hosted_zone_id>`, specify the public hosted zone for your domain.

For `<cluster_domain>`, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.

For `<hosted_zone_id>`, specify the public hosted zone ID for the load balancer that you obtained.

For `<external_ip>`, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.
2.10.22. Completing an AWS installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Amazon Web Service (AWS) user-provisioned infrastructure, monitor the deployment to completion.

Prerequisites

- You removed the bootstrap node for an OpenShift Container Platform cluster on user-provisioned AWS infrastructure.
- You installed the oc CLI.

Procedure

- From the directory that contains the installation program, complete the cluster installation:

  ```bash
  $ ./openshift-install --dir <installation_directory> wait-for install-complete
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Fe5en-ymBEc-Wt6NL"
INFO Time elapsed: 1s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
2.10.23. Logging in to the cluster by using the web console

The kubeadmin user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the kubeadmin user by using the OpenShift Container Platform web console.

Prerequisites

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

Procedure

1. Obtain the password for the kubeadmin user from the kubeadmin-password file on the installation host:

   $ cat <installation_directory>/auth/kubeadmin-password

   **NOTE**
   Alternatively, you can obtain the kubeadmin password from the <installation_directory>/openshift_install.log log file on the installation host.

2. List the OpenShift Container Platform web console route:

   $ oc get routes -n openshift-console | grep 'console-openshift'

   **NOTE**
   Alternatively, you can obtain the OpenShift Container Platform route from the <installation_directory>/openshift_install.log log file on the installation host.

   **Example output**

   console     console-openshift-console.apps.<cluster_name>.<base_domain>            console
   https   reencrypt/Redirect   None

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the kubeadmin user.

   **Additional resources**

   - See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

2.10.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

### Additional resources
- See About remote health monitoring for more information about the Telemetry service.

#### 2.10.25. Additional resources
- See Working with stacks in the AWS documentation for more information about AWS CloudFormation stacks.

#### 2.10.26. Next steps
- Validating an installation.
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If necessary, you can remove cloud provider credentials.

### 2.11. INSTALLING A CLUSTER ON AWS IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on Amazon Web Services (AWS) using infrastructure that you provide and an internal mirror of the installation release content.

**IMPORTANT**

While you can install an OpenShift Container Platform cluster by using mirrored installation release content, your cluster still requires Internet access to use the AWS APIs.

One way to create this infrastructure is to use the provided CloudFormation templates. You can modify the templates to customize your infrastructure or use the information that they contain to create AWS objects according to your company’s policies.

**IMPORTANT**

The steps for performing a user–provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several CloudFormation templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

#### 2.11.1. Prerequisites
- You created a mirror registry on your mirror host and obtained the imageContentSources data for your version of OpenShift Container Platform.
IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You reviewed details about the OpenShift Container Platform installation and update processes.
- You configured an AWS account to host the cluster.

IMPORTANT

If you have an AWS profile stored on your computer, it must not use a temporary session token that you generated while using a multi-factor authentication device. The cluster continues to use your current AWS credentials to create AWS resources for the entire life of the cluster, so you must use key-based, long-lived credentials. To generate appropriate keys, see Managing Access Keys for IAM Users in the AWS documentation. You can supply the keys when you run the installation program.

- You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix) in the AWS documentation.
- If you use a firewall and plan to use the Telemetry service, you configured the firewall to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

2.11.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.
IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

2.11.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The ClusterVersion status includes an Unable to retrieve available updates error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

2.11.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

2.11.4. Required AWS infrastructure components

To install OpenShift Container Platform on user-provisioned infrastructure in Amazon Web Services (AWS), you must manually create both the machines and their supporting infrastructure.

For more information about the integration testing for different platforms, see the OpenShift Container Platform 4.x Tested Integrations page.

By using the provided CloudFormation templates, you can create stacks of AWS resources that represent the following components:

- An AWS Virtual Private Cloud (VPC)
- Networking and load balancing components
• Security groups and roles
• An OpenShift Container Platform bootstrap node
• OpenShift Container Platform control plane nodes
• An OpenShift Container Platform compute node

Alternatively, you can manually create the components or you can reuse existing infrastructure that meets the cluster requirements. Review the CloudFormation templates for more details about how the components interrelate.

2.11.4.1. Cluster machines

You need **AWS::EC2::Instance** objects for the following machines:

- A bootstrap machine. This machine is required during installation, but you can remove it after your cluster deploys.
- Three control plane machines. The control plane machines are not governed by a machine set.
- Compute machines. You must create at least two compute machines, which are also known as worker machines, during installation. These machines are not governed by a machine set.

You can use the following instance types for the cluster machines with the provided CloudFormation templates.

**IMPORTANT**

If **m4** instance types are not available in your region, such as with eu-west-3, use **m5** types instead.

<table>
<thead>
<tr>
<th>Instance type</th>
<th>Bootstrap</th>
<th>Control plane</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>i3.large</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m4.large</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.xlarge</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m4.2xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.4xlarge</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m4.8xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.10xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>m4.16xlarge</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Instance type</td>
<td>Bootstrap</td>
<td>Control plane</td>
<td>Compute</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>m5.large</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>m5.xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m5.2xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m5.4xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m5.8xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m5.10xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m5.16xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>m6i.xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>c4.2xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>c4.4xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>c4.8xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>r4.large</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>r4.xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>r4.2xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>r4.4xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>r4.8xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>r4.16xlarge</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

You might be able to use other instance types that meet the specifications of these instance types.

2.11.4.2. Other infrastructure components

- A VPC
- DNS entries
- Load balancers (classic or network) and listeners
- A public and a private Route 53 zone
- Security groups
- IAM roles
- S3 buckets

If you are working in a disconnected environment or use a proxy, you cannot reach the public IP addresses for EC2 and ELB endpoints. To reach these endpoints, you must create a VPC endpoint and attach it to the subnet that the clusters are using. Create the following endpoints:

- `ec2.<region>.amazonaws.com`
- `elasticloadbalancing.<region>.amazonaws.com`
- `s3.<region>.amazonaws.com`

### Required VPC components

You must provide a suitable VPC and subnets that allow communication to your machines.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPC</td>
<td><strong>AWS::EC2::VPC</strong> &lt;br&gt; <strong>AWS::EC2::VPCEndpoint</strong></td>
<td>You must provide a public VPC for the cluster to use. The VPC uses an endpoint that references the route tables for each subnet to improve communication with the registry that is hosted in S3.</td>
</tr>
<tr>
<td>Public subnets</td>
<td><strong>AWS::EC2::Subnet</strong> &lt;br&gt; <strong>AWS::EC2::SubnetNetworkAclAssociation</strong></td>
<td>Your VPC must have public subnets for between 1 and 3 availability zones and associate them with appropriate Ingress rules.</td>
</tr>
<tr>
<td>Internet gateway</td>
<td><strong>AWS::EC2::InternetGateway</strong> &lt;br&gt; <strong>AWS::EC2::VPCGatewayAttachment</strong> &lt;br&gt; <strong>AWS::EC2::RouteTable</strong> &lt;br&gt; <strong>AWS::EC2::Route</strong> &lt;br&gt; <strong>AWS::EC2::SubnetRouteTableAssociation</strong> &lt;br&gt; <strong>AWS::EC2::NatGateway</strong> &lt;br&gt; <strong>AWS::EC2::EIP</strong></td>
<td>You must have a public Internet gateway, with public routes, attached to the VPC. In the provided templates, each public subnet has a NAT gateway with an EIP address. These NAT gateways allow cluster resources, like private subnet instances, to reach the Internet and are not required for some restricted network or proxy scenarios.</td>
</tr>
</tbody>
</table>
### Network access control

- **AWS::EC2::NetworkAcl**
- **AWS::EC2::NetworkAclEntry**

You must allow the VPC to access the following ports:

<table>
<thead>
<tr>
<th>Port</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Inbound HTTP traffic</td>
</tr>
<tr>
<td>443</td>
<td>Inbound HTTPS traffic</td>
</tr>
<tr>
<td>22</td>
<td>Inbound SSH traffic</td>
</tr>
<tr>
<td>1024 - 65535</td>
<td>Inbound ephemeral traffic</td>
</tr>
<tr>
<td>0 - 65535</td>
<td>Outbound ephemeral traffic</td>
</tr>
</tbody>
</table>

### Private subnets

- **AWS::EC2::Subnet**
- **AWS::EC2::RouteTable**
- **AWS::EC2::SubnetRouteTableAssociation**

Your VPC can have private subnets. The provided CloudFormation templates can create private subnets for between 1 and 3 availability zones. If you use private subnets, you must provide appropriate routes and tables for them.

### Required DNS and load balancing components

Your DNS and load balancer configuration needs to use a public hosted zone and can use a private hosted zone similar to the one that the installation program uses if it provisions the cluster’s infrastructure. You must create a DNS entry that resolves to your load balancer. An entry for `api.<cluster_name>.<domain>` must point to the external load balancer, and an entry for `api-int.<cluster_name>.<domain>` must point to the internal load balancer.

The cluster also requires load balancers and listeners for port 6443, which are required for the Kubernetes API and its extensions, and port 22623, which are required for the Ignition config files for new machines. The targets will be the control plane nodes (also known as the master nodes). Port 6443 must be accessible to both clients external to the cluster and nodes within the cluster. Port 22623 must be accessible to nodes within the cluster.

<table>
<thead>
<tr>
<th>Component</th>
<th>AWS type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>AWS::Route 53::HostedZone</td>
<td>The hosted zone for your internal DNS.</td>
</tr>
<tr>
<td>Component</td>
<td>AWS type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>etcd record sets</td>
<td>AWS::Route 53::RecordSet</td>
<td>The registration records for etcd for your control plane machines.</td>
</tr>
<tr>
<td>Public load balancer</td>
<td>AWS::Elastic LoadBalancingV2::LoadBalancer</td>
<td>The load balancer for your public subnets.</td>
</tr>
<tr>
<td>External API server record</td>
<td>AWS::Route 53::RecordSetGroup</td>
<td>Alias records for the external API server.</td>
</tr>
<tr>
<td>External listener</td>
<td>AWS::Elastic LoadBalancingV2::Listener</td>
<td>A listener on port 6443 for the external load balancer.</td>
</tr>
<tr>
<td>External target group</td>
<td>AWS::Elastic LoadBalancingV2::TargetGroup</td>
<td>The target group for the external load balancer.</td>
</tr>
<tr>
<td>Private load balancer</td>
<td>AWS::Elastic LoadBalancingV2::LoadBalancer</td>
<td>The load balancer for your private subnets.</td>
</tr>
<tr>
<td>Internal API server record</td>
<td>AWS::Route 53::RecordSetGroup</td>
<td>Alias records for the internal API server.</td>
</tr>
<tr>
<td>Internal listener</td>
<td>AWS::Elastic LoadBalancingV2::Listener</td>
<td>A listener on port 22623 for the internal load balancer.</td>
</tr>
<tr>
<td>Internal target group</td>
<td>AWS::Elastic LoadBalancingV2::TargetGroup</td>
<td>The target group for the internal load balancer.</td>
</tr>
<tr>
<td>Internal listener</td>
<td>AWS::Elastic LoadBalancingV2::Listener</td>
<td>A listener on port 6443 for the internal load balancer.</td>
</tr>
<tr>
<td>Component</td>
<td>AWS type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Internal target group</td>
<td>AWS::Elastic LoadBalancingV2::Target Group</td>
<td>The target group for the internal load balancer.</td>
</tr>
</tbody>
</table>

### Security groups

The control plane and worker machines require access to the following ports:

<table>
<thead>
<tr>
<th>Group</th>
<th>Type</th>
<th>IP Protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>6443</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22623</td>
</tr>
<tr>
<td>WorkerSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>icmp</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td>BootstrapSecurityGroup</td>
<td>AWS::EC2::Security Group</td>
<td>tcp</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tcp</td>
<td>19531</td>
</tr>
</tbody>
</table>

### Control plane Ingress

The control plane machines require the following Ingress groups. Each Ingress group is a AWS::EC2::SecurityGroupIngress resource.

<table>
<thead>
<tr>
<th>Ingress group</th>
<th>Description</th>
<th>IP protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterIngress Etcd</td>
<td>etcd</td>
<td>tcp</td>
<td>2379-2380</td>
</tr>
<tr>
<td>MasterIngress Vxlan</td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td>MasterIngress WorkerVxlan</td>
<td>Vxlan packets</td>
<td>udp</td>
<td>4789</td>
</tr>
<tr>
<td>MasterIngress Internal</td>
<td>Internal cluster communication and Kubernetes proxy metrics</td>
<td>tcp</td>
<td>9000-9999</td>
</tr>
<tr>
<td>Ingress group</td>
<td>Description</td>
<td>IP protocol</td>
<td>Port range</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>MasterIngress WorkerInternal</td>
<td>Internal cluster communication</td>
<td>tcp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>MasterIngress Kube</td>
<td>Kubernetes kubelet, scheduler and controller manager</td>
<td>tcp</td>
<td>10250 - 10259</td>
</tr>
<tr>
<td>MasterIngress WorkerKube</td>
<td>Kubernetes kubelet, scheduler and controller manager</td>
<td>tcp</td>
<td>10250 - 10259</td>
</tr>
<tr>
<td>MasterIngress IngressServices</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress WorkerIngress</td>
<td>Kubernetes Ingress services</td>
<td>tcp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress Geneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>MasterIngress WorkerGeneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>MasterIngress IpsecIKE</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>MasterIngress WorkerIpsecIKE</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>MasterIngress IpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>MasterIngress WorkerIpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>MasterIngress IpsecEsp</td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td>MasterIngress WorkerIpsecEsp</td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td>MasterIngress InternalUDP</td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
</tbody>
</table>
### MasterIngress

#### WorkerInternalUDP
- **Description**: Internal cluster communication
- **IP protocol**: udp
- **Port range**: 9000 - 9999

#### IngressServicesUDP
- **Description**: Kubernetes Ingress services
- **IP protocol**: udp
- **Port range**: 30000 - 32767

### Worker Ingress

The worker machines require the following Ingress groups. Each Ingress group is a `AWS::EC2::SecurityGroupIngress` resource.

<p>| Ingress group          | Description                                             | IP protocol | Port range     |
|------------------------|---------------------------------------------------------|-------------|----------------|----------------|
| <strong>WorkerIngress Vxlan</strong>| Vxlan packets                                           | udp         | 4789           |
| <strong>WorkerIngress Vxlan</strong>| Vxlan packets                                           | udp         | 4789           |
| <strong>WorkerIngress Internal</strong>| Internal cluster communication                          | tcp         | 9000 - 9999    |
| <strong>WorkerIngress Internal</strong>| Internal cluster communication                          | tcp         | 9000 - 9999    |
| <strong>WorkerIngress Kube</strong>   | Kubernetes kubelet, scheduler, and controller manager  | tcp         | 10250          |
| <strong>WorkerIngress Kube</strong>   | Kubernetes kubelet, scheduler, and controller manager  | tcp         | 10250          |
| <strong>WorkerIngress Services</strong>| Kubernetes Ingress services                            | tcp         | 30000 - 32767  |
| <strong>WorkerIngress Services</strong>| Kubernetes Ingress services                            | tcp         | 30000 - 32767  |</p>
<table>
<thead>
<tr>
<th>Ingress group</th>
<th>Description</th>
<th>IP protocol</th>
<th>Port range</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorkerIngress Geneve</td>
<td>Geneve packets</td>
<td>udp</td>
<td>6081</td>
</tr>
<tr>
<td>MasterGeneve</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorkerIngress IpsecIke</td>
<td>IPsec IKE packets</td>
<td>udp</td>
<td>500</td>
</tr>
<tr>
<td>MasterIpsecIke</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorkerIngress IpsecNat</td>
<td>IPsec NAT-T packets</td>
<td>udp</td>
<td>4500</td>
</tr>
<tr>
<td>MasterIpsecNat</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorkerIngress IpsecEsp</td>
<td>IPsec ESP packets</td>
<td>50</td>
<td>All</td>
</tr>
<tr>
<td>MasterIpsecEsp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorkerIngress InternalUDP</td>
<td>Internal cluster communication</td>
<td>udp</td>
<td>9000 - 9999</td>
</tr>
<tr>
<td>MasterInternalUDP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorkerIngress ServicesUDP</td>
<td>Kubernetes Ingress services</td>
<td>udp</td>
<td>30000 - 32767</td>
</tr>
<tr>
<td>MasterIngress ServicesUDP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Roles and instance profiles

You must grant the machines permissions in AWS. The provided CloudFormation templates grant the machines **Allow** permissions for the following **AWS::IAM::Role** objects and provide a **AWS::IAM::InstanceProfile** for each set of roles. If you do not use the templates, you can grant the machines the following broad permissions or the following individual permissions.
<table>
<thead>
<tr>
<th>Role</th>
<th>Effect</th>
<th>Action</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master</td>
<td>Allow</td>
<td>ec2:*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>elasticloadbalancing:*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>iam:PassRole</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>s3:GetObject</td>
<td>*</td>
</tr>
<tr>
<td>Worker</td>
<td>Allow</td>
<td>ec2:Describe*</td>
<td>*</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>Allow</td>
<td>ec2:Describe*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>ec2:AttachVolume</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>Allow</td>
<td>ec2:DetachVolume</td>
<td>*</td>
</tr>
</tbody>
</table>

2.11.4.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

2.11.4.4. Required AWS permissions

**NOTE**

Your IAM user must have the permission `tag:GetResources` in the region **us-east-1** to delete the base cluster resources. As part of the AWS API requirement, the OpenShift Container Platform installation program performs various actions in this region.

When you attach the `AdministratorAccess` policy to the IAM user that you create in Amazon Web Services (AWS), you grant that user all of the required permissions. To deploy all components of an OpenShift Container Platform cluster, the IAM user requires the following permissions:

**Example 2.31. Required EC2 permissions for installation**

- `tag:TagResources`
- `tag:UntagResources`
- `ec2:AllocateAddress`
- `ec2:AssociateAddress`
CHAPTER 2. INSTALLING ON AWS

- `ec2:AuthorizeSecurityGroupEgress`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:CopyImage`
- `ec2:CreateNetworkInterface`
- `ec2:AttachNetworkInterface`
- `ec2:CreateSecurityGroup`
- `ec2:CreateTags`
- `ec2:CreateVolume`
- `ec2:DeleteSecurityGroup`
- `ec2:DeleteSnapshot`
- `ec2:DeleteTags`
- `ec2:DeregisterImage`
- `ec2:DescribeAccountAttributes`
- `ec2:DescribeAddresses`
- `ec2:DescribeAvailabilityZones`
- `ec2:DescribeDhcpOptions`
- `ec2:DescribeImages`
- `ec2:DescribeInstanceAttribute`
- `ec2:DescribeInstanceCreditSpecifications`
- `ec2:DescribeInstances`
- `ec2:DescribeInternetGateways`
- `ec2:DescribeKeyPairs`
- `ec2:DescribeNatGateways`
- `ec2:DescribeNetworkAcls`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribePrefixLists`
- `ec2:DescribeRegions`
- `ec2:DescribeRouteTables`
- `ec2:DescribeSecurityGroups`
Example 2.32. Required permissions for creating network resources during installation

- ec2:AssociateDhcpOptions
- ec2:AssociateRouteTable
- ec2:AttachInternetGateway
- ec2:CreateDhcpOptions
- ec2:CreateInternetGateway
- ec2:CreateNatGateway
- ec2:CreateRoute
- ec2:CreateRouteTable
- ec2:CreateSubnet
- ec2:CreateVpc
- ec2:CreateVpcEndpoint
- ec2:ModifySubnetAttribute
• ec2:ModifyVpcAttribute

NOTE
If you use an existing VPC, your account does not require these permissions for creating network resources.

Example 2.33. Required Elastic Load Balancing permissions (ELB) for installation

• elasticloadbalancing:AddTags
• elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
• elasticloadbalancing:AttachLoadBalancerToSubnets
• elasticloadbalancing:ConfigureHealthCheck
• elasticloadbalancing:CreateLoadBalancer
• elasticloadbalancing:CreateLoadBalancerListeners
• elasticloadbalancing:DeleteLoadBalancer
• elasticloadbalancing:DeregisterInstancesFromLoadBalancer
• elasticloadbalancing:DescribeInstanceHealth
• elasticloadbalancing:DescribeLoadBalancerAttributes
• elasticloadbalancing:DescribeLoadBalancers
• elasticloadbalancing:DescribeTags
• elasticloadbalancing:ModifyLoadBalancerAttributes
• elasticloadbalancing:RegisterInstancesWithLoadBalancer
• elasticloadbalancing:SetLoadBalancerPoliciesOfListener

Example 2.34. Required Elastic Load Balancing permissions (ELBv2) for installation

• elasticloadbalancing:AddTags
• elasticloadbalancing:CreateListener
• elasticloadbalancing:CreateLoadBalancer
• elasticloadbalancing:CreateTargetGroup
• elasticloadbalancing:DeleteLoadBalancer
• elasticloadbalancing:DeregisterTargets
• elasticloadbalancing:DescribeListeners
- elasticloadbalancing:DescribeLoadBalancerAttributes
- elasticloadbalancing:DescribeLoadBalancers
- elasticloadbalancing:DescribeTargetGroupAttributes
- elasticloadbalancing:DescribeTargetHealth
- elasticloadbalancing:ModifyLoadBalancerAttributes
- elasticloadbalancing:ModifyTargetGroup
- elasticloadbalancing:ModifyTargetGroupAttributes
- elasticloadbalancing:RegisterTargets

Example 2.35. Required IAM permissions for installation

- iam:AddRoleToInstanceProfile
- iam:CreateInstanceProfile
- iam:CreateRole
- iam:DeleteInstanceProfile
- iam:DeleteRole
- iam:DeleteRolePolicy
- iam:GetInstanceProfile
- iam:GetRole
- iam:GetRolePolicy
- iam:GetUser
- iam:ListInstanceProfilesForRole
- iam:ListRoles
- iam:ListUsers
- iam:PassRole
- iam:PutRolePolicy
- iam:RemoveRoleFromInstanceProfile
- iam:SimulatePrincipalPolicy
- iam:TagRole
NOTE

If you have not created an elastic load balancer (ELB) in your AWS account, the IAM user also requires the `iam:CreateServiceLinkedRole` permission.

Example 2.36. Required Route 53 permissions for installation

- `route53:ChangeResourceRecordSets`
- `route53:ChangeTagsForResource`
- `route53:CreateHostedZone`
- `route53:DeleteHostedZone`
- `route53:GetChange`
- `route53:GetHostedZone`
- `route53:GetHostedZone`
- `route53:GetHostedZone`
- `route53:ListHostedZones`
- `route53:ListHostedZonesByName`
- `route53:ListResourceRecordSets`
- `route53:ListTagsForResource`
- `route53:UpdateHostedZoneComment`

Example 2.37. Required S3 permissions for installation

- `s3:CreateBucket`
- `s3:DeleteBucket`
- `s3:GetAccelerateConfiguration`
- `s3:GetBucketAcl`
- `s3:GetBucketCors`
- `s3:GetBucketLocation`
- `s3:GetBucketLogging`
- `s3:GetBucketObjectLockConfiguration`
- `s3:GetBucketReplication`
- `s3:GetBucketRequestPayment`
- `s3:GetBucketTagging`
- `s3:GetBucketVersioning`
• s3:GetBucketWebsite
• s3:GetEncryptionConfiguration
• s3:GetLifecycleConfiguration
• s3:GetReplicationConfiguration
• s3:ListBucket
• s3:PutBucketAcl
• s3:PutBucketTagging
• s3:PutEncryptionConfiguration

Example 2.38. S3 permissions that cluster Operators require

• s3:DeleteObject
• s3:GetObject
• s3:GetObjectAcl
• s3:GetObjectTagging
• s3:GetObjectVersion
• s3:PutObject
• s3:PutObjectAcl
• s3:PutObjectTagging

Example 2.39. Required permissions to delete base cluster resources

• autoscaling:DescribeAutoScalingGroups
• ec2:DeleteNetworkInterface
• ec2:DeleteVolume
• elasticloadbalancing:DeleteTargetGroup
• elasticloadbalancing:DescribeTargetGroups
• iam:DeleteAccessKey
• iam:DeleteUser
• iam:ListAttachedRolePolicies
• iam:ListInstanceProfiles
• iam:ListRolePolicies
Example 2.40. Required permissions to delete network resources

- `ec2:DeleteDhcpOptions`
- `ec2:DeleteInternetGateway`
- `ec2:DeleteNatGateway`
- `ec2:DeleteRoute`
- `ec2:DeleteRouteTable`
- `ec2:DeleteSubnet`
- `ec2:DeleteVpc`
- `ec2:DeleteVpcEndpoints`
- `ec2:DetachInternetGateway`
- `ec2:DisassociateRouteTable`
- `ec2:ReplaceRouteTableAssociation`

**NOTE**

If you use an existing VPC, your account does not require these permissions to delete network resources.

Example 2.41. Additional IAM and S3 permissions that are required to create manifests

- `iam:DeleteAccessKey`
- `iam:DeleteUser`
- `iam:DeleteUserPolicy`
- `iam:GetUserPolicy`
- `iam:ListAccessKeys`
- `iam:PutUserPolicy`
- `iam:TagUser`
- `iam:GetUserPolicy`
- `iam:ListAccessKeys`
- `s3:PutBucketPublicAccessBlock`
- `s3:GetBucketPublicAccessBlock`
- `s3:PutLifecycleConfiguration`
- `s3:HeadBucket`
- `s3:ListBucketMultipartUploads`
- `s3:AbortMultipartUpload`

**NOTE**

If you are managing your cloud provider credentials with mint mode, the IAM user also requires the `iam:CreateAccessKey` and `iam:CreateUser` permissions.

**Example 2.42. Optional permission for quota checks for installation**

- `servicequotas:ListAWSDefaultServiceQuotas`

**2.11.5. Generating an SSH private key and adding it to the agent**

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```
Specifying the SSH Key

1. Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**
If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**
   
   
   Agent pid 31874

   **NOTE**
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**
   
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   **NOTE**
   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

   **Next steps**
   * When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

2.11.6. Creating the installation files for AWS

To install OpenShift Container Platform on Amazon Web Services (AWS) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the `install-config.yaml` file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate `var` partition during the preparation phases of installation.
2.11.6.1. Optional: Creating a separate /var partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the /var partition or a subdirectory of /var. For example:

- /var/lib/containers: Holds container-related content that can grow as more images and containers are added to a system.
- /var/lib/etcd: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- /var: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a /var directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because /var must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate /var partition by creating a machine config that is inserted during the openshift-install preparation phases of an OpenShift Container Platform installation.

IMPORTANT

If you follow the steps to create a separate /var partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```
   $ mkdir $HOME/clusterconfig
   ```

2. Run openshift-install to create a set of files in the manifest and openshift subdirectories. Answer the system questions as you are prompted:

   ```
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ```

   Example output

   ```
   ? SSH Public Key ...
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: $HOME/clusterconfig/manifests and $HOME/clusterconfig/openshift
   ```

3. Optional: Confirm that the installation program created manifests in the clusterconfig/openshift directory:
Example output

$ ls $HOME/clusterconfig/openshift/

99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...

4. Create a **MachineConfig** object and add it to a file in the **openshift** directory. For example, name the file **98-var-partition.yaml**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the /var directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>
          partitions:
            - label: var
              startMiB: <partition_start_offset>
              sizeMiB: <partition_size>
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
  systemd:
    units:
      - name: var.mount
        enabled: true
        contents:
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota
          [Install]
          WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. The disk device name to the name of the storage device on the **worker** systems.
3. The partition start offset.
4. The partition size.
5. The partition options.

CHAPTER 2. INSTALLING ON AWS
When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the size of the data partition in mebibytes.

3. The size of the data partition in mebibytes.

4. The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-lib-containers.mount`.

5. The `prjquota` mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

```bash
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 2.11.6.2. Creating the installation configuration file

Generate and customize the installation configuration file that the installation program needs to deploy your cluster.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program for user-provisioned infrastructure and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

- You checked that you are deploying your cluster to a region with an accompanying Red Hat Enterprise Linux CoreOS (RHCOS) AMI published by Red Hat. If you are deploying to a region that requires a custom AMI, such as an AWS GovCloud region, you must create the `install-config.yaml` file manually.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>
   ```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select `aws` as the platform to target.

iii. If you do not have an AWS profile stored on your computer, enter the AWS access key ID and secret access key for the user that you configured to run the installation program.

**NOTE**

The AWS access key ID and secret access key are stored in `~/.aws/credentials` in the home directory of the current user on the installation host. You are prompted for the credentials by the installation program if the credentials for the exported profile are not present in the file. Any credentials that you provide to the installation program are stored in the file.

iv. Select the AWS region to deploy the cluster to.

v. Select the base domain for the Route 53 service that you configured for your cluster.

vi. Enter a descriptive name for your cluster.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Edit the `install-config.yaml` file to provide the additional information that is required for an installation in a restricted network.

a. Update the `pullSecret` value to contain the authentication information for your registry:

```
pullSecret: {'"auths":{"<local_registry>" : {"auth": "<credentials>" , "email": "you@example.com"}}}
```
For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

b. Add the `additionalTrustBundle` parameter and value. The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

```
additionalTrustBundle: |
-----BEGIN CERTIFICATE-----
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
-----END CERTIFICATE-----
```

c. Add the image content resources:

```
imageContentSources:
- mirrors:
  - `<local_registry>/<local_repository_name>/release`
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - `<local_registry>/<local_repository_name>/release`
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

Use the `imageContentSources` section from the output of the command to mirror the repository or the values that you used when you mirrored the content from the media that you brought into your restricted network.

d. Optional: Set the publishing strategy to `Internal`:

```
publish: Internal
```

By setting this option, you create an internal Ingress Controller and a private load balancer.

3. Optional: Back up the `install-config.yaml` file.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

**Additional resources**

- See [Configuration and credential file settings](#) in the AWS documentation for more information about AWS profile and credential configuration.

**2.11.6.3. Configuring the cluster-wide proxy during installation**

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**
You have an existing `install-config.yaml` file.

You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

If your cluster is on AWS, you added the `ec2.<region>.amazonaws.com`, `elasticloadbalancing.<region>.amazonaws.com`, and `s3.<region>.amazonaws.com` endpoints to your VPC endpoint. These endpoints are required to complete requests from the nodes to the AWS EC2 API. Because the proxy works on the container level, not the node level, you must route these requests to the AWS EC2 API through the AWS private network. Adding the public IP address of the EC2 API to your allowlist in your proxy server is not sufficient.

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
   noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...
   ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For
   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The

   ```
additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

**NOTE**

Only the Proxy object named cluster is supported, and no additional proxies can be created.

### 2.11.6.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.

- You created the install-config.yaml installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:
$ ./openshift-install create manifests --dir <installation_directory> ①

Example output

INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift

① For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines*.yaml

   By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset*.yaml

   Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

5. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

   ```yaml
   apiVersion: config.openshift.io/v1
   kind: DNS
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     baseDomain: example.openshift.com
     privateZone:
       id: mycluster-100419-private-zone
     publicZone:
       id: example.openshift.com
   status: {}
   ```
Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

6. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```
   ├── auth
   │   └── kubeadm-password
   │       └── kubeconfig
   │           └── bootstrap.ign
   │                   └── master.ign
   │                           └── metadata.json
   │                                   └── worker.ign
   ```

2.11.7. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Amazon Web Services (AWS). The infrastructure name is also used to locate the appropriate AWS resources during an OpenShift Container Platform installation. The provided CloudFormation templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  ```bash
  $ jq -r .infraID <installation_directory>/metadata.json
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```bash
openshift-vw9j6
```
The output of this command is your cluster name and a random string.

2.11.8. Creating a VPC in AWS

You must create a Virtual Private Cloud (VPC) in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements, including VPN and route tables.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the VPC.

NOTE

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.

Procedure

1. Create a JSON file that contains the parameter values that the template requires:

   ```json
   [  
     {  
       "ParameterKey": "VpcCidr",  
       "ParameterValue": "10.0.0.0/16"  
     },  
     {  
       "ParameterKey": "AvailabilityZoneCount",  
       "ParameterValue": "1"  
     },  
     {  
       "ParameterKey": "SubnetBits",  
       "ParameterValue": "12"  
     }  
   ]
   ```

   1. The CIDR block for the VPC.
   2. Specify a CIDR block in the format `x.x.x.x/16-24`.
   3. The number of availability zones to deploy the VPC in.
   4. Specify an integer between 1 and 3.
The size of each subnet in each availability zone.

Specify an integer between 5 and 13, where 5 is \(\frac{27}{27}\) and 13 is \(\frac{19}{19}\).

2. Copy the template from the CloudFormation template for the VPC section of this topic and save it as a YAML file on your computer. This template describes the VPC that your cluster requires.

3. Launch the CloudFormation template to create a stack of AWS resources that represent the VPC:

   **IMPORTANT**
   
   You must enter the command on a single line.

   ```
   $ aws cloudformation create-stack --stack-name <name> 
   --template-body file://<template>.yaml 
   --parameters file://<parameters>.json 
   ```

   - `<name>` is the name for the CloudFormation stack, such as `cluster-vpc`. You need the name of this stack if you remove the cluster.
   - `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
   - `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.

   **Example output**

   ```
   arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
   ```

4. Confirm that the template components exist:

   ```
   $ aws cloudformation describe-stacks --stack-name <name>
   ```

   After the `StackStatus` displays `CREATE_COMPLETE`, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VpcId</td>
<td>The ID of your VPC.</td>
</tr>
<tr>
<td>PublicSubnetIds</td>
<td>The IDs of the new public subnets.</td>
</tr>
<tr>
<td>PrivateSubnetIds</td>
<td>The IDs of the new private subnets.</td>
</tr>
</tbody>
</table>
2.11.8.1. CloudFormation template for the VPC

You can use the following CloudFormation template to deploy the VPC that you need for your OpenShift Container Platform cluster.

Example 2.43. CloudFormation template for the VPC

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^((0-9][1-9][0-9]*[2-9]0-4][0-9]*[2-9]25[0-5])\.)3\{0-9][1-9][0-9]*[2-9]25[0-5]\}2[0-9][0-9]*[2-9]25[0-5]\}($
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
    Type: Number

Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label: "Network Configuration"
    Parameters:
      - VpcCidr
      - SubnetBits
    - Label: "Availability Zones"
      Parameters:
        - AvailabilityZoneCount

ParameterLabels:
  AvailabilityZoneCount:
    default: "Availability Zone Count"
  VpcCidr:
    default: "VPC CIDR"
  SubnetBits:
    default: "Bits Per Subnet"

Conditions:
DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
```
DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
VPC:
  Type: "AWS::EC2::VPC"
  Properties:
    EnableDnsSupport: "true"
    EnableDnsHostnames: "true"
    CidrBlock: !Ref VpcCidr
PublicSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 0
    - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PublicSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
InternetGateway:
  Type: "AWS::EC2::InternetGateway"
GatewayToInternet:
  Type: "AWS::EC2::VPCGatewayAttachment"
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref InternetGateway
PublicRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PublicRoute:
  Type: "AWS::EC2::Route"
  DependsOn: GatewayToInternet
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
SubnetId: !Ref PublicSubnet
RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PublicSubnet2
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
  Condition: DoAz3
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP
        - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
Properties:
  VpcId: !Ref VPC
  CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
  AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"

PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC

PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2

NAT2:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP2
        - AllocationId
    SubnetId: !Ref PublicSubnet2

EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc

Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2

PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"

PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3

NAT3:
  DependsOn:
    - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
        - EIP3
        - AllocationId
    SubnetId: !Ref PublicSubnet3

EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc

Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      - Ref: PrivateRouteTable3
      - DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      - Ref: NAT3

S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Principal: '*'
          Action:
            - '*'
          Resource:
            - '*'
        RouteTableIds:
          - !Ref PublicRouteTable
          - !Ref PrivateRouteTable
          - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
          - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
        ServiceName: !Join
          - ''
          - com.amazonaws.
          - !Ref 'AWS::Region'
          - .s3
      VpcId: !Ref VPC

Outputs:
2.11.9. Creating networking and load balancing components in AWS

You must configure networking and classic or network load balancing in Amazon Web Services (AWS) that your OpenShift Container Platform cluster can use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the networking and load balancing components that your OpenShift Container Platform cluster requires. The template also creates a hosted zone and subnet tags.

You can run the template multiple times within a single Virtual Private Cloud (VPC).

**NOTE**

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

**Procedure**

1. Obtain the hosted zone ID for the Route 53 base domain that you specified in the `install-config.yaml` file for your cluster. You can obtain details about your hosted zone by running the following command:
For the `<route53_domain>`, specify the Route 53 base domain that you used when you generated the `install-config.yaml` file for the cluster.

Example output

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

In the example output, the hosted zone ID is `Z21IXYZABCZ2A4`.

2. Create a JSON file that contains the parameter values that the template requires:

```
[
  {
    "ParameterKey": "ClusterName",  
    "ParameterValue": "mycluster"  
  },
  {
    "ParameterKey": "InfrastructureName",  
    "ParameterValue": "mycluster-<random_string>"  
  },
  {
    "ParameterKey": "HostedZoneId",  
    "ParameterValue": "<random_string>"  
  },
  {
    "ParameterKey": "HostedZoneName",  
    "ParameterValue": "example.com"  
  },
  {
    "ParameterKey": "PublicSubnets",  
    "ParameterValue": "subnet-<random_string>"  
  },
  {
    "ParameterKey": "PrivateSubnets",  
    "ParameterValue": "subnet-<random_string>"  
  },
  {
    "ParameterKey": "VpcId",  
    "ParameterValue": "vpc-<random_string>"  
  }
]
```

1. A short, representative cluster name to use for hostnames, etc.

2. Specify the cluster name that you used when you generated the `install-config.yaml` file for the cluster.
The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

4 Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

5 The Route 53 public zone ID to register the targets with.

6 Specify the Route 53 public zone ID, which has a format similar to Z21IXYZABCZ2A4. You can obtain this value from the AWS console.

7 The Route 53 zone to register the targets with.

8 Specify the Route 53 base domain that you used when you generated the `install-config.yaml` file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.

9 The public subnets that you created for your VPC.

10 Specify the `PublicSubnetIds` value from the output of the CloudFormation template for the VPC.

11 The private subnets that you created for your VPC.

12 Specify the `PrivateSubnetIds` value from the output of the CloudFormation template for the VPC.

13 The VPC that you created for the cluster.

14 Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

3. Copy the template from the CloudFormation template for the network and load balancers section of this topic and save it as a YAML file on your computer. This template describes the networking and load balancing objects that your cluster requires.

   **IMPORTANT**

   If you are deploying your cluster to an AWS government region, you must update the `InternalApiServerRecord` in the CloudFormation template to use CNAME records. Records of type ALIAS are not supported for AWS government regions.

4. Launch the CloudFormation template to create a stack of AWS resources that provide the networking and load balancing components:

   **IMPORTANT**

   You must enter the command on a single line.

   ```
   $ aws cloudformation create-stack --stack-name <name>  
   --template-body file://<template>.yaml  
   --parameters file://<parameters>.json  
   --capabilities CAPABILITY_NAMED_IAM
   ```
<name> is the name for the CloudFormation stack, such as cluster-dns. You need the name of this stack if you remove the cluster.

<template> is the relative path to and name of the CloudFormation template YAML file that you saved.

<parameters> is the relative path to and name of the CloudFormation parameters JSON file.

You must explicitly declare the CAPABILITY_NAMED_IAM capability because the provided template creates some AWS::IAM::Role resources.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

After the StackStatus displays CREATE_COMPLETE, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrivateHostedZoneId</td>
<td>Hosted zone ID for the private DNS.</td>
</tr>
<tr>
<td>ExternalApiLoadBalancerName</td>
<td>Full name of the external API load balancer.</td>
</tr>
<tr>
<td>InternalApiLoadBalancerName</td>
<td>Full name of the internal API load balancer.</td>
</tr>
<tr>
<td>ApiServerDnsName</td>
<td>Full hostname of the API server.</td>
</tr>
<tr>
<td>RegisterNlbIpTargetLambda</td>
<td>Lambda ARN useful to help register/deregister IP targets for these load balancers.</td>
</tr>
<tr>
<td>ExternalApiTargetGroupArn</td>
<td>ARN of external API target group.</td>
</tr>
</tbody>
</table>
### 2.11.9.1. CloudFormation template for the network and load balancers

You can use the following CloudFormation template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster.

**Example 2.44. CloudFormation template for the network and load balancers**

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
ClusterName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, representative cluster name to use for host names and other identifying names.
  Type: String
InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
  Type: String
HostedZoneId:
  Description: The Route53 public zone ID to register the targets with, such as Z21IXYZABCZ2A4.
  Type: String
HostedZoneName:
  Description: The Route53 zone to register the targets with, such as example.com. Omit the trailing period.
  Type: String
Default: "example.com"
PublicSubnets:
  Description: The internet-facing subnets.
  Type: List<AWS::EC2::Subnet::Id>
PrivateSubnets:
  Description: The internal subnets.
  Type: List<AWS::EC2::Subnet::Id>
VpcId:
```

<table>
<thead>
<tr>
<th>InternalApiTargetGroupArn</th>
<th>ARN of internal API target group.</th>
</tr>
</thead>
<tbody>
<tr>
<td>InternalServiceTargetGroupArn</td>
<td>ARN of internal service target group.</td>
</tr>
</tbody>
</table>
Description: The VPC-scoped resources will belong to this VPC.
Type: AWS::EC2::VPC::Id

Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label: default: "Cluster Information"
    Parameters:
    - ClusterName
    - InfrastructureName
  - Label: default: "Network Configuration"
    Parameters:
    - VpcId
    - PublicSubnets
    - PrivateSubnets
  - Label: default: "DNS"
    Parameters:
    - HostedZoneName
    - HostedZoneId

ParameterLabels:
  ClusterName:
  default: "Cluster Name"
  InfrastructureName:
  default: "Infrastructure Name"
  VpcId:
  default: "VPC ID"
  PublicSubnets:
  default: "Public Subnets"
  PrivateSubnets:
  default: "Private Subnets"
  HostedZoneName:
  default: "Public Hosted Zone Name"
  HostedZoneId:
  default: "Public Hosted Zone ID"

Resources:
ExtApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
  Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
  IpAddressType: ipv4
  Subnets: !Ref PublicSubnets
  Type: network

IntApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
  Name: !Join ["-", [!Ref InfrastructureName, "int"]]
  Scheme: internal
  IpAddressType: ipv4
  Subnets: !Ref PrivateSubnets
  Type: network
IntDns:
Type: "AWS::Route53::HostedZone"
Properties:
  HostedZoneConfig:
    Comment: "Managed by CloudFormation"
    Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
  HostedZoneTags:
    - Key: Name
      Value: !Join ["-", [!Ref InfrastructureName, "int"]]
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
VPCs:
  - VPCId: !Ref VpcId
  VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:
Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref HostedZoneId
  RecordSets:
    - Name: !Join [
        ".", [
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
    AliasTarget:
      HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
      DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
Type: AWS::Route53::RecordSetGroup
Properties:
  Comment: Alias record for the API server
  HostedZoneId: !Ref IntDns
  RecordSets:
    - Name: !Join [
        ".", [
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
    AliasTarget:
      HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
      DNSName: !GetAtt IntApiElb.DNSName

- Name: !Join [
    ".", [
      ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
    ]
  Type: A
  AliasTarget:
    HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
    DNSName: !GetAtt IntApiElb.DNSName
ExternalApiListener:
Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
  DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
LoadBalancerArn:
  Ref: ExtApiElb
Port: 6443
Protocol: TCP

ExternalApiTargetGroup:
Type: AWS::ElasticLoadBalancingV2::TargetGroup
Properties:
  HealthCheckIntervalSeconds: 10
  HealthCheckPath: "/readyz"
  HealthCheckPort: 6443
  HealthCheckProtocol: HTTPS
  HealthyThresholdCount: 2
  UnhealthyThresholdCount: 2
  Port: 6443
  Protocol: TCP
  TargetType: ip
  VpcId:
    Ref: VpcId
  TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalApiListener:
Type: AWS::ElasticLoadBalancingV2::Listener
Properties:
  DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
LoadBalancerArn:
  Ref: IntApiElb
Port: 6443
Protocol: TCP

InternalApiTargetGroup:
Type: AWS::ElasticLoadBalancingV2::TargetGroup
Properties:
  HealthCheckIntervalSeconds: 10
  HealthCheckPath: "/readyz"
  HealthCheckPort: 6443
  HealthCheckProtocol: HTTPS
  HealthyThresholdCount: 2
  UnhealthyThresholdCount: 2
  Port: 6443
  Protocol: TCP
  TargetType: ip
  VpcId:
    Ref: VpcId
TargetGroupAttributes:
  - Key: deregistration_delay.timeout_seconds
    Value: 60

InternalServiceInternalListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
      - Type: forward
    TargetGroupArn:
      Ref: InternalServiceTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
    Port: 22623
    Protocol: TCP

InternalServiceTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: /healthz
    HealthCheckPort: 22623
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 22623
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

RegisterTargetLambdaIamRole:
  Type: AWS::IAM::Role
  Properties:
    RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "lambda.amazonaws.com"
          Action:
            - "sts:AssumeRole"
          Path: "/"
        Policies:
          - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
            PolicyDocument:
              Version: "2012-10-17"
              Statement:
                - Effect: "Allow"
                  Action: [
"elasticloadbalancing:RegisterTargets",
"elasticloadbalancing:DeregisterTargets",
]
Resource: !Ref InternalApiTargetGroup
- Effect: "Allow"
Action:
  [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
  ]
Resource: !Ref InternalServiceTargetGroup
- Effect: "Allow"
Action:
  [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets",
  ]
Resource: !Ref ExternalApiTargetGroup

RegisterNlbIpTargets:
  Type: "AWS::Lambda::Function"
Properties:
    Handler: "index.handler"
    Role:
      Fn::GetAtt:
        - "RegisterTargetLambdaIamRole"
        - "Arn"
    Code:
      ZipFile:
        import json
        import boto3
        import cfnresponse
        def handler(event, context):
          elb = boto3.client('elbv2')
          if event['RequestType'] == 'Delete':
            elb.deregister_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'], Targets=[{'Id': event['ResourceProperties']['TargetIp']}]])
          else:
            event['RequestType'] == 'Create':
            elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'], Targets=[{'Id': event['ResourceProperties']['TargetIp']}]])
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
            event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
        Runtime: "python3.7"
        Timeout: 120

RegisterSubnetTagsLambdamRole:
  Type: AWS::IAM::Role
Properties:
    RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
- "lambda.amazonaws.com"
  
Action:
  - "sts:AssumeRole"

Path: "/"

Policies:
- PolicyName: !Join [ ",", [!Ref InfrastructureName, "subnet-tagging-policy"]]

PolicyDocument:
  
Version: "2012-10-17"

Statement:
- Effect: "Allow"

Action:

["ec2:DeleteTags",
"ec2:CreateTags"
]

Resource: "arn:aws:ec2:*:*:subnet/*"

- Effect: "Allow"

Action:

["ec2:DescribeSubnets",
"ec2:DescribeTags"
]

Resource: "*"

RegisterSubnetTags:

Type: "AWS::Lambda::Function"

Properties:

Handler: "index.handler"

Role:

Fn::GetAtt:
  - "RegisterSubnetTagsLambdaIamRole"
  - "Arn"

Code:

ZipFile:

```
import json
import boto3
import cfnresponse

def handler(event, context):
    ec2_client = boto3.client('ec2')
    if event['RequestType'] == 'Delete':
        for subnet_id in event['ResourceProperties']['Subnets']:
            ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
                event['ResourceProperties']['InfrastructureName']}]));
    elif event['RequestType'] == 'Create':
        for subnet_id in event['ResourceProperties']['Subnets']:
            ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
                event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'})
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
                event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])

Runtime: "python3.7"
Timeout: 120
```

RegisterPublicSubnetTags:

Type: Custom::SubnetRegister

Properties:
IMPORTANT

If you are deploying your cluster to an AWS government region, you must update the `InternalApiServerRecord` to use CNAME records. Records of type ALIAS are not supported for AWS government regions. For example:

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

Addional resources

- See Listing public hosted zones in the AWS documentation for more information about listing public hosted zones.
2.11.10. Creating security group and roles in AWS

You must create security groups and roles in Amazon Web Services (AWS) for your OpenShift Container Platform cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the security groups and roles that your OpenShift Container Platform cluster requires.

**NOTE**

If you do not use the provided CloudFormation template to create your AWS infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.

**Procedure**

1. Create a JSON file that contains the parameter values that the template requires:

   ```json
   [
   {
     "ParameterKey": "InfrastructureName",  
     "ParameterValue": "mycluster-<random_string>"  
   },
   {
     "ParameterKey": "VpcCidr",  
     "ParameterValue": "10.0.0.0/16"  
   },
   {
     "ParameterKey": "PrivateSubnets",  
     "ParameterValue": "subnet-<random_string>"  
   },
   {
     "ParameterKey": "VpcId",  
     "ParameterValue": "vpc-<random_string>"  
   }
   ]
   ```

   1. The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
   2. Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`. 
3. The CIDR block for the VPC.

4. Specify the CIDR block parameter that you used for the VPC that you defined in the form `x.x.x.x/16-24`.

5. The private subnets that you created for your VPC.

6. Specify the `PrivateSubnetIds` value from the output of the CloudFormation template for the VPC.

7. The VPC that you created for the cluster.

8. Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

2. Copy the template from the CloudFormation template for security objects section of this topic and save it as a YAML file on your computer. This template describes the security groups and roles that your cluster requires.

3. Launch the CloudFormation template to create a stack of AWS resources that represent the security groups and roles:

   **IMPORTANT**
   
   You must enter the command on a single line.

   ```
   $ aws cloudformation create-stack --stack-name <name>  
   --template-body file://<template>.yaml  
   --parameters file://<parameters>.json  
   --capabilities CAPABILITY_NAMED_IAM
   ```

   - `<name>` is the name for the CloudFormation stack, such as `cluster-sec`. You need the name of this stack if you remove the cluster.
   - `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
   - `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.
   - You must explicitly declare the `CAPABILITY_NAMED_IAM` capability because the provided template creates some `AWS::IAM::Role` and `AWS::IAM::InstanceProfile` resources.

### Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. Confirm that the template components exist:

   ```
   $ aws cloudformation describe-stacks --stack-name <name>
   ```
After the StackStatus displays CREATE_COMPLETE, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterSecurityGroupId</td>
<td>Master Security Group ID</td>
</tr>
<tr>
<td>WorkerSecurityGroupId</td>
<td>Worker Security Group ID</td>
</tr>
<tr>
<td>MasterInstanceProfile</td>
<td>Master IAM Instance Profile</td>
</tr>
<tr>
<td>WorkerInstanceProfile</td>
<td>Worker IAM Instance Profile</td>
</tr>
</tbody>
</table>

2.11.10.1. CloudFormation template for security objects

You can use the following CloudFormation template to deploy the security objects that you need for your OpenShift Container Platform cluster.

Example 2.45. CloudFormation template for security objects

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
InfrastructureName:
  AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
  Type: String
VpcCidr:
  AllowedPattern: ^([0-9]{1,3}\.){3}[0-9]{1,3}[/16-24]$
  ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
  Default: 10.0.0.0/16
  Description: CIDR block for VPC.
  Type: String
VpcId:
  Description: The VPC-scoped resources will belong to this VPC.
  Type: AWS::EC2::VPC::Id
PrivateSubnets:
  Description: The internal subnets.
  Type: List<AWS::EC2::Subnet::Id>
```
Metadata:
AWS::CloudFormation::Interface:
  ParameterGroups:
  - Label:
    - default: "Cluster Information"
  Parameters:
  - InfrastructureName
  - Label:
    - default: "Network Configuration"
  Parameters:
  - VpcId
  - VpcCidr
  - PrivateSubnets
  ParameterLabels:
  InfrastructureName:
  - default: "Infrastructure Name"
  VpcId:
  - default: "VPC ID"
  VpcCidr:
  - default: "VPC CIDR"
  PrivateSubnets:
  - default: "Private Subnets"

Resources:
MasterSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
  GroupDescription: Cluster Master Security Group
  SecurityGroupIngress:
  - IpProtocol: icmp
    FromPort: 0
    ToPort: 0
    CidrIp: !Ref VpcCidr
  - IpProtocol: tcp
    FromPort: 22
    ToPort: 22
    CidrIp: !Ref VpcCidr
  - IpProtocol: tcp
    ToPort: 6443
    FromPort: 6443
    CidrIp: !Ref VpcCidr
  - IpProtocol: tcp
    FromPort: 22623
    ToPort: 22623
    CidrIp: !Ref VpcCidr
  VpcId: !Ref VpcId

WorkerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
  GroupDescription: Cluster Worker Security Group
  SecurityGroupIngress:
  - IpProtocol: icmp
    FromPort: 0
    ToPort: 0
CidrIp: !Ref VpcCidr
- IpProtocol: tcp
  FromPort: 22
  ToPort: 22
CidrIp: !Ref VpcCidr
VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

MasterIngressWorkerGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp
MasterIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressWorkerInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

MasterIngressWorkerInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

MasterIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes kubelet, scheduler and controller manager
    FromPort: 10250
    ToPort: 10259
    IpProtocol: tcp

MasterIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes kublet, scheduler and controller manager
FromPort: 10250
ToPort: 10259
IpProtocol: tcp

MasterIngressIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

MasterIngressWorkerIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: tcp

MasterIngressIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt MasterSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

WorkerIngressVxlan:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Vxlan packets
FromPort: 4789
ToPort: 4789
IpProtocol: udp

WorkerIngressMasterVxlan:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
- Description: Vxlan packets
- FromPort: 4789
- ToPort: 4789
- IpProtocol: udp

WorkerIngressGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
- Description: Geneve packets
- FromPort: 6081
- ToPort: 6081
- IpProtocol: udp

WorkerIngressMasterGeneve:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
- Description: Geneve packets
- FromPort: 6081
- ToPort: 6081
- IpProtocol: udp

WorkerIngressInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
- Description: Internal cluster communication
- FromPort: 9000
- ToPort: 9999
- IpProtocol: tcp

WorkerIngressMasterInternal:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
- Description: Internal cluster communication
- FromPort: 9000
- ToPort: 9999
- IpProtocol: tcp

WorkerIngressInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
- GroupId: !GetAtt WorkerSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
- Description: Internal cluster communication
- FromPort: 9000
WorkerIngressMasterInternalUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Internal cluster communication
  FromPort: 9000
  ToPort: 9999
  IpProtocol: udp

WorkerIngressKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes secure kubelet port
  FromPort: 10250
  ToPort: 10250
  IpProtocol: tcp

WorkerIngressWorkerKube:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Internal Kubernetes communication
  FromPort: 10250
  ToPort: 10250
  IpProtocol: tcp

WorkerIngressIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: tcp

WorkerIngressMasterIngressServices:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: tcp

WorkerIngressIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
WorkerIngressMasterIngressServicesUDP:
Type: AWS::EC2::SecurityGroupIngress
Properties:
  GroupId: !GetAtt WorkerSecurityGroup.GroupId
  SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
  Description: Kubernetes ingress services
  FromPort: 30000
  ToPort: 32767
  IpProtocol: udp

WorkerSecurityGroup:

MasterIamRole:
Type: AWS::IAM::Role
Properties:
AssumeRolePolicyDocument:
  Version: "2012-10-17"
Statement:
  - Effect: "Allow"
    Principal:
      Service:
        - "ec2.amazonaws.com"
    Action:
      - "sts:AssumeRole"
  Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: "Allow"
            Action:
              - "ec2:AttachVolume"
              - "ec2:AuthorizeSecurityGroupIngress"
              - "ec2:CreateSecurityGroup"
              - "ec2:CreateTags"
              - "ec2:CreateVolume"
              - "ec2:DeleteSecurityGroup"
              - "ec2:DeleteVolume"
              - "ec2:Describe**"
              - "ec2:DetachVolume"
              - "ec2:ModifyInstanceAttribute"
              - "ec2:ModifyVolume"
              - "ec2:RevokeSecurityGroupIngress"
              - "elasticloadbalancing:AddTags"
              - "elasticloadbalancing:AttachLoadBalancerToSubnets"
              - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
              - "elasticloadbalancing:CreateListener"
              - "elasticloadbalancing:CreateLoadBalancer"
              - "elasticloadbalancing:CreateLoadBalancerPolicy"
              - "elasticloadbalancing:CreateLoadBalancerListeners"
              - "elasticloadbalancing:CreateTargetGroup"
- "elasticloadbalancing:ConfigureHealthCheck"
- "elasticloadbalancing:DeleteListener"
- "elasticloadbalancing:DeleteLoadBalancer"
- "elasticloadbalancing:DeleteLoadBalancerListeners"
- "elasticloadbalancing:DeleteTargetGroup"
- "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
- "elasticloadbalancing:DeregisterTargets"
- "elasticloadbalancing:Describe**"
- "elasticloadbalancing:DetachLoadBalancerFromSubnets"
- "elasticloadbalancing:ModifyListener"
- "elasticloadbalancing:ModifyLoadBalancerAttributes"
- "elasticloadbalancing:ModifyTargetGroup"
- "elasticloadbalancing:ModifyTargetGroupAttributes"
- "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
- "elasticloadbalancing:RegisterTargets"
- "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
- "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
- "kms:DescribeKey"

Resource: "***"

MasterInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
    - Ref: "MasterIamRole"

WorkerIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
        Action:
        - "sts:AssumeRole"
    Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Action:
          - "ec2:DescribeInstances"
          - "ec2:DescribeRegions"
          Resource: "***"

WorkerInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Roles:
    - Ref: "WorkerIamRole"

Outputs:
2.11.11. RHCOS AMIs for the AWS infrastructure

Red Hat provides Red Hat Enterprise Linux CoreOS (RHCOS) AMIs valid for the various Amazon Web Services (AWS) zones you can specify for your OpenShift Container Platform nodes.

NOTE
You can also install to regions that do not have a RHCOS AMI published by importing your own AMI.

Table 2.33. RHCOS AMIs

<table>
<thead>
<tr>
<th>AWS zone</th>
<th>AWS AMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>af-south-1</td>
<td>ami-09921c9c1c36e695c</td>
</tr>
<tr>
<td>ap-east-1</td>
<td>ami-01ee8446e9af6b197</td>
</tr>
<tr>
<td>ap-northeast-1</td>
<td>ami-04e5b5722a55846ea</td>
</tr>
<tr>
<td>ap-northeast-2</td>
<td>ami-0fdc25c8a0273a742</td>
</tr>
<tr>
<td>ap-south-1</td>
<td>ami-09e3deb397cc526a8</td>
</tr>
<tr>
<td>ap-southeast-1</td>
<td>ami-0630e03f75e02eec4</td>
</tr>
<tr>
<td>ap-southeast-2</td>
<td>ami-069450613262ba03c</td>
</tr>
<tr>
<td>ca-central-1</td>
<td>ami-012518cdbd3057dfd</td>
</tr>
<tr>
<td>eu-central-1</td>
<td>ami-0bd7175ff5b1aef0c</td>
</tr>
<tr>
<td>eu-north-1</td>
<td>ami-06c9ec42d0a839ad2</td>
</tr>
</tbody>
</table>
### AWS zone | AWS AMI
---|---
eu-south-1 | ami-0614d7440a0363d71
eu-west-1 | ami-01b89df58b5d4d5fa
eu-west-2 | ami-06f6e31dd554f89d
eu-west-3 | ami-0dc82e2517ded15a1
me-south-1 | ami-07d181e3aa0f76067
sa-east-1 | ami-0cd44e6dd20e6c7fa
us-east-1 | ami-04a16d506e5b0e246
us-east-2 | ami-0a1f868ad58ea59a7
us-west-1 | ami-0a65d76e3a6f6622f
us-west-2 | ami-0dd9008abadc519f1

### 2.11.12. Creating the bootstrap node in AWS

You must create the bootstrap node in Amazon Web Services (AWS) to use during OpenShift Container Platform cluster initialization.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources. The stack represents the bootstrap node that your OpenShift Container Platform installation requires.

**NOTE**

If you do not use the provided CloudFormation template to create your bootstrap node, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
Procedure

1. Provide a location to serve the `bootstrap.ign` Ignition config file to your cluster. This file is located in your installation directory. One way to do this is to create an S3 bucket in your cluster’s region and upload the Ignition config file to it.

   **IMPORTANT**

   The provided CloudFormation Template assumes that the Ignition config files for your cluster are served from an S3 bucket. If you choose to serve the files from another location, you must modify the templates.

   **IMPORTANT**

   If you are deploying to a region that has endpoints that differ from the AWS SDK, or you are providing your own custom endpoints, you must use a presigned URL for your S3 bucket instead of the `s3://` schema.

   **NOTE**

   The bootstrap Ignition config file does contain secrets, like X.509 keys. The following steps provide basic security for the S3 bucket. To provide additional security, you can enable an S3 bucket policy to allow only certain users, such as the OpenShift IAM user, to access objects that the bucket contains. You can avoid S3 entirely and serve your bootstrap Ignition config file from any address that the bootstrap machine can reach.

   a. Create the bucket:

      ```
      $ aws s3 mb s3://<cluster-name>-infra
      
      <cluster-name>-infra is the bucket name. When creating the `install-config.yaml` file, replace `<cluster-name>` with the name specified for the cluster.
      ```

   b. Upload the `bootstrap.ign` Ignition config file to the bucket:

      ```
      $ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign
      
      For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
      ```

   c. Verify that the file uploaded:

      ```
      $ aws s3 ls s3://<cluster-name>-infra/
      
      Example output
      
      2019-04-03 16:15:16  314878 bootstrap.ign
      ```

2. Create a JSON file that contains the parameter values that the template requires:
[{
  "ParameterKey": "InfrastructureName", 1
  "ParameterValue": "mycluster-<random_string>" 2
},
{
  "ParameterKey": "RhcosAmi", 3
  "ParameterValue": "ami-<random_string>" 4
},
{
  "ParameterKey": "AllowedBootstrapSshCidr", 5
  "ParameterValue": "0.0.0.0/0" 6
},
{
  "ParameterKey": "PublicSubnet", 7
  "ParameterValue": "subnet-<random_string>" 8
},
{
  "ParameterKey": "MasterSecurityGroupId", 9
  "ParameterValue": "sg-<random_string>" 10
},
{
  "ParameterKey": "VpcId", 11
  "ParameterValue": "vpc-<random_string>" 12
},
{
  "ParameterKey": "BootstrapIgnitionLocation", 13
  "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
},
{
  "ParameterKey": "AutoRegisterELB", 15
  "ParameterValue": "yes" 16
},
{
  "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
  "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
},
{
  "ParameterKey": "ExternalApiTargetGroupArn", 19
  "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
},
{
  "ParameterKey": "InternalApiTargetGroupArn", 21
  "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
},
{
  "ParameterKey": "InternalServiceTargetGroupArn", 23
  "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.

Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the bootstrap node.

Specify a valid `AWS::EC2::Image::Id` value.

CIDR block to allow SSH access to the bootstrap node.

Specify a CIDR block in the format `x.x.x.x/16-24`.

The public subnet that is associated with your VPC to launch the bootstrap node into.

Specify the `PublicSubnetIds` value from the output of the CloudFormation template for the VPC.

The master security group ID (for registering temporary rules)

Specify the `MasterSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

The VPC created resources will belong to.

Specify the `VpcId` value from the output of the CloudFormation template for the VPC.

Location to fetch bootstrap Ignition config file from.

Specify the S3 bucket and file name in the form `s3://<bucket_name>/bootstrap.ign`.

Whether or not to register a network load balancer (NLB).

Specify `yes` or `no`. If you specify `yes`, you must provide a Lambda Amazon Resource Name (ARN) value.

The ARN for NLB IP target registration lambda group.

Specify the `RegisterNlbIpTargetsLambda` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for external API load balancer target group.

Specify the `ExternalApiTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

The ARN for internal API load balancer target group.
Specify the `InternalApiTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

Specify the `InternalServiceTargetGroupArn` value from the output of the CloudFormation template for DNS and load balancing. Use `arn:aws-us-gov` if deploying the cluster to an AWS GovCloud region.

3. Copy the template from the **CloudFormation template for the bootstrap machine** section of this topic and save it as a YAML file on your computer. This template describes the bootstrap machine that your cluster requires.

4. Launch the CloudFormation template to create a stack of AWS resources that represent the bootstrap node:

   **IMPORTANT**

   You must enter the command on a single line.

   ```bash
   $ aws cloudformation create-stack --stack-name <name>  
   --template-body file://<template>.yaml  
   --parameters file://<parameters>.json  
   --capabilities CAPABILITY_NAMED_IAM
   ```

   - `<name>` is the name for the CloudFormation stack, such as `cluster-bootstrap`. You need the name of this stack if you remove the cluster.
   - `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
   - `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.
   - You must explicitly declare the `CAPABILITY_NAMED_IAM` capability because the provided template creates some `AWS::IAM::Role` and `AWS::IAM::InstanceProfile` resources.

**Example output**

```
ar:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

5. Confirm that the template components exist:

   ```bash
   $ aws cloudformation describe-stacks --stack-name <name>
   ```

   After the **StackStatus** displays **CREATE_COMPLETE**, the output displays values for the following parameters. You must provide these parameter values to the other CloudFormation templates that you run to create your cluster:
<table>
<thead>
<tr>
<th><strong>Bootstrap InstanceId</strong></th>
<th>The bootstrap Instance ID.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bootstrap PublicIp</strong></td>
<td>The bootstrap node public IP address.</td>
</tr>
<tr>
<td><strong>Bootstrap PrivateIp</strong></td>
<td>The bootstrap node private IP address.</td>
</tr>
</tbody>
</table>

### 2.11.12.1. CloudFormation template for the bootstrap machine

You can use the following CloudFormation template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster.

#### Example 2.46. CloudFormation template for the bootstrap machine

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^((\d{1,3}\.\d{1,3}\d{1,3}\d{1,3}(/\d{0,3}))*$)
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
```

---

**CHAPTER 2. INSTALLING ON AWS**

433
Default: "yes"
AllowedValues:
- "yes"
- "no"
Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
Type: String

RegisterNlbIPTargetsLambdaArn:
Description: ARN for NLB IP target registration lambda.
Type: String

ExternalApiTargetGroupArn:
Description: ARN for external API load balancer target group.
Type: String

InternalApiTargetGroupArn:
Description: ARN for internal API load balancer target group.
Type: String

InternalServiceTargetGroupArn:
Description: ARN for internal service load balancer target group.
Type: String

Metadata:
AWS::CloudFormation::Interface:
ParameterGroups:
- Label: "Cluster Information"
  Parameters:
  - InfrastructureName
- Label: "Host Information"
  Parameters:
  - RhcosAmi
  - BootstrapIgnitionLocation
  - MasterSecurityGroupId
- Label: "Network Configuration"
  Parameters:
  - VpcId
  - AllowedBootstrapSshCidr
  - PublicSubnet
- Label: "Load Balancer Automation"
  Parameters:
  - AutoRegisterELB
  - RegisterNlbIPTargetsLambdaArn
  - ExternalApiTargetGroupArn
  - InternalApiTargetGroupArn
  - InternalServiceTargetGroupArn
ParameterLabels:
InfrastructureName:
  default: "Infrastructure Name"
VpcId:
  default: "VPC ID"
AllowedBootstrapSshCidr:
  default: "Allowed SSH Source"
PublicSubnet:
  default: "Public Subnet"
RhcosAmi:
default: "Red Hat Enterprise Linux CoreOS AMI ID"
BootstrapIgnitionLocation:
default: "Bootstrap Ignition Source"
MasterSecurityGroupId:
default: "Master Security Group ID"
AutoRegisterELB:
default: "Use Provided ELB Automation"

Conditions:
DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
BootstrapIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service:
              - "ec2.amazonaws.com"
          Action:
            - "sts:AssumeRole"
          Path: "/
        Policies:
          - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
            PolicyDocument:
              Version: "2012-10-17"
              Statement:
                - Effect: "Allow"
                  Action: "ec2:Describe*"
                  Resource: "*"
                - Effect: "Allow"
                  Action: "ec2:AttachVolume"
                  Resource: "*"
                - Effect: "Allow"
                  Action: "ec2:DetachVolume"
                  Resource: "*"
                - Effect: "Allow"
                  Action: "s3:GetObject"
                  Resource: "*"

BootstrapInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Path: "/
    Roles:
      - Ref: "BootstrapIamRole"

BootstrapSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Bootstrap Security Group
    SecurityGroupIngress:
      - IpProtocol: tcp
FromPort: 22
ToPort: 22
CidrIp: !Ref AllowedBootstrapSshCidr
- IpProtocol: tcp
  ToPort: 19531
  FromPort: 19531
  CidrIp: 0.0.0.0/0
VpcId: !Ref VpcId

BootstrapInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    IAMInstanceProfile: !Ref BootstrapInstanceProfile
    InstanceType: "i3.large"
  NetworkInterfaces:
    - AssociatePublicIpAddress: "true"
      DeviceIndex: "0"
      GroupSet:
        - !Ref "BootstrapSecurityGroup"
        - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "PublicSubnet"
      UserData:
        Fn::Base64: !Sub
          - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
          - {S3Loc: !Ref BootstrapIgnitionLocation}

RegisterBootstrapApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

RegisterBootstrapInternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

Outputs:
BootstrapInstanceId:
  Description: Bootstrap Instance ID.
Additional resources

- See [RHCOS AMIs for the AWS infrastructure](#) for details about the Red Hat Enterprise Linux CoreOS (RHCOS) AMIs for the AWS zones.

### 2.11.13. Creating the control plane machines in AWS

You must create the control plane machines in Amazon Web Services (AWS) that your cluster will use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent the control plane nodes.

**IMPORTANT**

The CloudFormation template creates a stack that represents three control plane nodes.

**NOTE**

If you do not use the provided CloudFormation template to create your control plane nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
- You created and configured a VPC and associated subnets in AWS.
- You created and configured DNS, load balancers, and listeners in AWS.
- You created the security groups and roles required for your cluster in AWS.
- You created the bootstrap machine.

**Procedure**

1. Create a JSON file that contains the parameter values that the template requires:

```json
Value: !Ref BootstrapInstance

BootstrapPublicIp:
  Description: The bootstrap node public IP address.
  Value: !GetAtt BootstrapInstance.PublicIp

BootstrapPrivateIp:
  Description: The bootstrap node private IP address.
  Value: !GetAtt BootstrapInstance.PrivateIp
```
[{
  "ParameterKey": "InfrastructureName", 1
  "ParameterValue": "mycluster-<random_string>" 2
},
{
  "ParameterKey": "RhcosAmi", 3
  "ParameterValue": "ami-<random_string>" 4
},
{
  "ParameterKey": "AutoRegisterDNS", 5
  "ParameterValue": "yes" 6
},
{
  "ParameterKey": "PrivateHostedZoneId", 7
  "ParameterValue": "<random_string>" 8
},
{
  "ParameterKey": "PrivateHostedZoneName", 9
  "ParameterValue": "mycluster.example.com" 10
},
{
  "ParameterKey": "Master0Subnet", 11
  "ParameterValue": "subnet-<random_string>" 12
},
{
  "ParameterKey": "Master1Subnet", 13
  "ParameterValue": "subnet-<random_string>" 14
},
{
  "ParameterKey": "Master2Subnet", 15
  "ParameterValue": "subnet-<random_string>" 16
},
{
  "ParameterKey": "MasterSecurityGroupId", 17
  "ParameterValue": "sg-<random_string>" 18
},
{
  "ParameterKey": "IgnitionLocation", 19
  "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master" 20
},
{
  "ParameterKey": "CertificateAuthorities", 21
  "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz=" 22
},
{
  "ParameterKey": "MasterInstanceProfileName", 23
  "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
},
{
  "ParameterKey": "MasterInstanceType", 25
}]
"ParameterValue": "m4.xlarge" 26
},
{
 "ParameterKey": "AutoRegisterELB", 27
 "ParameterValue": "yes" 28
},
{
 "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 29
 "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 30
},
{
 "ParameterKey": "ExternalApiTargetGroupArn", 31
 "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
},
{
 "ParameterKey": "InternalApiTargetGroupArn", 33
 "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
},
{
 "ParameterKey": "InternalServiceTargetGroupArn", 35
 "ParameterValue": "arn:aws:elasticloadbalancing:<region>:<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
}
]

1. The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
2. Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.
3. Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the control plane machines.
4. Specify an AWS::EC2::Image::Id value.
5. Whether or not to perform DNS etcd registration.
6. Specify yes or no. If you specify yes, you must provide hosted zone information.
7. The Route 53 private zone ID to register the etcd targets with.
8. Specify the PrivateHostedZoneId value from the output of the CloudFormation template for DNS and load balancing.
9. The Route 53 zone to register the targets with.
10. Specify `<cluster_name>.<domain_name>` where `<domain_name>` is the Route 53 base domain that you used when you generated `install-config.yaml` file for the cluster. Do not include the trailing period (.) that is displayed in the AWS console.
11. A subnet, preferably private, to launch the control plane machines on.
Specify a subnet from the `PrivateSubnets` value from the output of the CloudFormation template for DNS and load balancing.

The master security group ID to associate with control plane nodes (also known as the master nodes).

Specify the `MasterSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

The location to fetch control plane Ignition config file from.

Specify the generated Ignition config file location, `https://api-int.<cluster_name>.<domain_name>:22623/config/master`.

The base64 encoded certificate authority string to use.

Specify the value from the `master.ign` file that is in the installation directory. This value is the long string with the format `data:text/plain;charset=utf-8;base64,ABC...xYz==`.

The IAM profile to associate with control plane nodes.

Specify the `MasterInstanceProfile` parameter value from the output of the CloudFormation template for the security group and roles.

The type of AWS instance to use for the control plane machines.

Allowed values:

- `m4.xlarge`
- `m4.2xlarge`
- `m4.4xlarge`
- `m4.8xlarge`
- `m4.10xlarge`
- `m4.16xlarge`
- `m5.xlarge`
- `m5.2xlarge`
- `m5.4xlarge`
- `m5.8xlarge`
- `m5.10xlarge`
- `m5.16xlarge`
- `m6i.xlarge`
- `c4.2xlarge`
- `c4.4xlarge`
• c4.8xlarge
• r4.xlarge
• r4.2xlarge
• r4.4xlarge
• r4.8xlarge
• r4.16xlarge

IMPORTANT

If m4 instance types are not available in your region, such as with eu-west-3, specify an m5 type, such as m5.xlarge, instead.

27. Whether or not to register a network load balancer (NLB).
28. Specify yes or no. If you specify yes, you must provide a Lambda Amazon Resource Name (ARN) value.
29. The ARN for NLB IP target registration lambda group.
30. Specify the RegisterNlbIpTargetsLambda value from the output of the CloudFormation template for DNS and load balancing. Use arn:aws-us-gov if deploying the cluster to an AWS GovCloud region.
31. The ARN for external API load balancer target group.
32. Specify the ExternalApiTargetGroupArn value from the output of the CloudFormation template for DNS and load balancing. Use arn:aws-us-gov if deploying the cluster to an AWS GovCloud region.
33. The ARN for internal API load balancer target group.
34. Specify the InternalApiTargetGroupArn value from the output of the CloudFormation template for DNS and load balancing. Use arn:aws-us-gov if deploying the cluster to an AWS GovCloud region.
35. The ARN for internal service load balancer target group.
36. Specify the InternalServiceTargetGroupArn value from the output of the CloudFormation template for DNS and load balancing. Use arn:aws-us-gov if deploying the cluster to an AWS GovCloud region.

2. Copy the template from the CloudFormation template for control plane machines section of this topic and save it as a YAML file on your computer. This template describes the control plane machines that your cluster requires.

3. If you specified an m5 instance type as the value for MasterInstanceType, add that instance type to the MasterInstanceType.AllowedValues parameter in the CloudFormation template.

4. Launch the CloudFormation template to create a stack of AWS resources that represent the control plane nodes:
IMPORTANT

You must enter the command on a single line.

```
$ aws cloudformation create-stack --stack-name <name> 1
   --template-body file://<template>.yaml 2
   --parameters file://<parameters>.json 3
```

1. `<name>` is the name for the CloudFormation stack, such as `cluster-control-plane`. You need the name of this stack if you remove the cluster.

2. `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.

3. `<parameters>` is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```

NOTE

The CloudFormation template creates a stack that represents three control plane nodes.

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

2.11.13.1. CloudFormation template for control plane machines

You can use the following CloudFormation template to deploy the control plane machines that you need for your OpenShift Container Platform cluster.

Example 2.47. CloudFormation template for control plane machines

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
InfrastructureName:
   AllowedPattern: ^([a-zA-Z][a-zA-Z0-9-]{0,26})$
   MaxLength: 27
   MinLength: 1
   ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
   Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
   Type: String
RhcosAmi:
```
Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.

Type: AWS::EC2::Image::Id

AutoRegisterDNS:
   Default: "yes"
   AllowedValues:
       - "yes"
       - "no"
   Description: Do you want to invoke DNS etcd registration, which requires Hosted Zone information?

Type: String

PrivateHostedZoneId:
   Description: The Route53 private zone ID to register the etcd targets with, such as Z21IXYZABCZ2A4.

Type: String

PrivateHostedZoneName:
   Description: The Route53 zone to register the targets with, such as cluster.example.com. Omit the trailing period.

Type: String

Master0Subnet:
   Description: The subnets, recommend private, to launch the master nodes into.

Type: AWS::EC2::Subnet::Id

Master1Subnet:
   Description: The subnets, recommend private, to launch the master nodes into.

Type: AWS::EC2::Subnet::Id

Master2Subnet:
   Description: The subnets, recommend private, to launch the master nodes into.

Type: AWS::EC2::Subnet::Id

MasterSecurityGroupId:
   Description: The master security group ID to associate with master nodes.

Type: AWS::EC2::SecurityGroup::Id

IgnitionLocation:
   Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
   Description: Ignition config file location.

Type: String

CertificateAuthorities:
   Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
   Description: Base64 encoded certificate authority string to use.

Type: String

MasterInstanceProfileName:
   Description: IAM profile to associate with master nodes.

Type: String

MasterInstanceType:
   Default: m5.xlarge

Type: String

AllowedValues:
   - "m4.xlarge"
   - "m4.2xlarge"
   - "m4.4xlarge"
   - "m4.10xlarge"
   - "m4.16xlarge"
   - "m5.xlarge"
   - "m5.2xlarge"
   - "m5.4xlarge"
   - "m5.8xlarge"
   - "m5.12xlarge"
   - "m5.16xlarge"
AutoRegisterELB:
Default: "yes"
AllowedValues:
- "yes"
- "no"
Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
Type: String
RegisterNlbIpTargetsLambdaArn:
Description: ARN for NLB IP target registration lambda. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
Type: String
ExternalApiTargetGroupArn:
Description: ARN for external API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
Type: String
InternalApiTargetGroupArn:
   Description: ARN for internal API load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
   Type: String
InternalServiceTargetGroupArn:
   Description: ARN for internal service load balancer target group. Supply the value from the cluster infrastructure or select "no" for AutoRegisterELB.
   Type: String

Metadata:
AWS::CloudFormation::Interface:
   ParameterGroups:
   - Label: "Cluster Information"
     Parameters:
     - InfrastructureName
   - Label: "Host Information"
     Parameters:
     - MasterInstanceType
     - RhcosAmi
     - IgnitionLocation
     - CertificateAuthorities
     - MasterSecurityGroupId
     - MasterInstanceProfileName
   - Label: "Network Configuration"
     Parameters:
     - VpcId
     - AllowedBootstrapSshCidr
     - Master0Subnet
     - Master1Subnet
     - Master2Subnet
   - Label: "DNS"
     Parameters:
     - AutoRegisterDNS
     - PrivateHostedZoneName
     - PrivateHostedZoneId
   - Label: "Load Balancer Automation"
     Parameters:
     - AutoRegisterELB
     - RegisterNlbIpTargetsLambdaArn
     - ExternalApiTargetGroupArn
     - InternalApiTargetGroupArn
     - InternalServiceTargetGroupArn
ParameterLabels:
InfrastructureName:
   default: "Infrastructure Name"
VpcId:
   default: "VPC ID"
Master0Subnet:
   default: "Master-0 Subnet"
Master1Subnet:
   default: "Master-1 Subnet"
Master2Subnet:
  default: "Master-2 Subnet"
MasterInstanceType:
  default: "Master Instance Type"
MasterInstanceProfileName:
  default: "Master Instance Profile Name"
RhcosAmi:
  default: "Red Hat Enterprise Linux CoreOS AMI ID"
BootstrapIgnitionLocation:
  default: "Master Ignition Source"
CertificateAuthorities:
  default: "Ignition CA String"
MasterSecurityGroupId:
  default: "Master Security Group ID"
AutoRegisterDNS:
  default: "Use Provided DNS Automation"
AutoRegisterELB:
  default: "Use Provided ELB Automation"
PrivateHostedZoneName:
  default: "Private Hosted Zone Name"
PrivateHostedZoneId:
  default: "Private Hosted Zone ID"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
  DoDns: !Equals ["yes", !Ref AutoRegisterDNS]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
        - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
        - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master0Subnet"
      UserData:
        Fn::Base64: !Sub
          - '{"ignition":{"config":{"merge":{{"source":"${SOURCE}"}}},"security":{"tls":
              {"certificateAuthorities":{{"source":"${CA_BUNDLE}"}}},"version":"3.1.0"}}}
            - {
                SOURCE: !Ref IgnitionLocation,
                CA_BUNDLE: !Ref CertificateAuthorities,
            }
      Tags:
        - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"
RegisterMaster0:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalApiTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalApiTargetGroupArn
  TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalServiceTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalServiceTargetGroupArn
  TargetIp: !GetAtt Master0.PrivateIp

Master1:
Type: AWS::EC2::Instance
Properties:
  ImageId: !Ref RhcosAmi
  BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
  IamInstanceProfile: !Ref MasterInstanceProfileName
  InstanceType: !Ref MasterInstanceType
  NetworkInterfaces:
    - AssociatePublicIpAddress: "false"
      DeviceIndex: "0"
      GroupSet:
        - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "Master1Subnet"
  UserData:
    Fn::Base64: !Sub
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
          {"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}
        - 
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
  Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

RegisterMaster1:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master1.PrivateIp

Master2:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
          - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
    UserData:
      Fn::Base64:
        - !Sub
          - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
            {"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]},"version":"3.1.0"}}}
            - '
              SOURCE: !Ref IgnitionLocation,
              CA_BUNDLE: !Ref CertificateAuthorities,
            
            Tag:
              - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
              Value: "shared"

RegisterMaster2:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
TargetArn: !Ref ExternalApiTargetGroupArn
TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalApiTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalApiTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalServiceTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
  TargetArn: !Ref InternalServiceTargetGroupArn
  TargetIp: !GetAtt Master2.PrivateIp

EtcdSrvRecords:
Condition: DoDns
Type: AWS::Route53::RecordSet
Properties:
  HostedZoneId: !Ref PrivateHostedZoneId
  Name: !Join [“.”, [“_etcd-server-ssl._tcp”, !Ref PrivateHostedZoneName]]
  ResourceRecords:
  - !Join [“”, [
    [0 10 2380", !Join [“.”, [”etcd-0", !Ref PrivateHostedZoneName]]],
  ]
  - !Join [“”, [
    [0 10 2380", !Join [“.”, [”etcd-1", !Ref PrivateHostedZoneName]]],
  ]
  - !Join [“”, [
    [0 10 2380", !Join [“.”, [”etcd-2", !Ref PrivateHostedZoneName]]],
  ]
  TTL: 60
  Type: SRV

Etcd0Record:
Condition: DoDns
Type: AWS::Route53::RecordSet
Properties:
  HostedZoneId: !Ref PrivateHostedZoneId
  Name: !Join [“.”, [”etcd-0", !Ref PrivateHostedZoneName]]
  ResourceRecords:
  - !GetAtt Master0.PrivateIp
  TTL: 60
  Type: A

Etcd1Record:
Condition: DoDns
Type: AWS::Route53::RecordSet
Properties:
2.11.14. Creating the worker nodes in AWS

You can create worker nodes in Amazon Web Services (AWS) for your cluster to use.

You can use the provided CloudFormation template and a custom parameter file to create a stack of AWS resources that represent a worker node.

**IMPORTANT**

The CloudFormation template creates a stack that represents one worker node. You must create a stack for each worker node.

**NOTE**

If you do not use the provided CloudFormation template to create your worker nodes, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- You configured an AWS account.
- You added your AWS keys and region to your local AWS profile by running `aws configure`.
- You generated the Ignition config files for your cluster.
You created and configured a VPC and associated subnets in AWS.

You created and configured DNS, load balancers, and listeners in AWS.

You created the security groups and roles required for your cluster in AWS.

You created the bootstrap machine.

You created the control plane machines.

**Procedure**

1. Create a JSON file that contains the parameter values that the CloudFormation template requires:

   ```json
   [
   {
   "ParameterKey": "InfrastructureName", 1
   "ParameterValue": "mycluster-<random_string>" 2
   },
   {
   "ParameterKey": "RhcosAmi", 3
   "ParameterValue": "ami-<random_string>" 4
   },
   {
   "ParameterKey": "Subnet", 5
   "ParameterValue": "subnet-<random_string>" 6
   },
   {
   "ParameterKey": "WorkerSecurityGroupId", 7
   "ParameterValue": "sg-<random_string>" 8
   },
   {
   "ParameterKey": "IgnitionLocation", 9
   "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker" 10
   },
   {
   "ParameterKey": "CertificateAuthorities", 11
   "ParameterValue": "" 12
   },
   {
   "ParameterKey": "WorkerInstanceProfileName", 13
   "ParameterValue": "" 14
   },
   {
   "ParameterKey": "WorkerInstanceType", 15
   "ParameterValue": "m4.large" 16
   }
   ]
   ```

   **notes**
   
   1. The name for your cluster infrastructure that is encoded in your Ignition config files for the cluster.
Specify the infrastructure name that you extracted from the Ignition config file metadata, which has the format `<cluster-name>-<random-string>`.

Current Red Hat Enterprise Linux CoreOS (RHCOS) AMI to use for the worker nodes.

Specify an `AWS::EC2::Image::Id` value.

A subnet, preferably private, to launch the worker nodes on.

Specify a subnet from the `PrivateSubnets` value from the output of the CloudFormation template for DNS and load balancing.

The worker security group ID to associate with worker nodes.

Specify the `WorkerSecurityGroupId` value from the output of the CloudFormation template for the security group and roles.

The location to fetch bootstrap Ignition config file from.

Specify the generated Ignition config location, `https://api-int.<cluster_name>.<domain_name>:22623/config/worker`.

Base64 encoded certificate authority string to use.

Specify the value from the `worker.ign` file that is in the installation directory. This value is the long string with the format `data:text/plain;charset=utf-8;base64,ABC…xYz==`.

The IAM profile to associate with worker nodes.

Specify the `WorkerInstanceProfile` parameter value from the output of the CloudFormation template for the security group and roles.

The type of AWS instance to use for the control plane machines.

**Allowed values:**

- `m4.large`
- `m4.xlarge`
- `m4.2xlarge`
- `m4.4xlarge`
- `m4.8xlarge`
- `m4.10xlarge`
- `m4.16xlarge`
- `m5.large`
- `m5.xlarge`
- `m5.2xlarge`
- `m5.4xlarge`
• m5.8xlarge
• m5.10xlarge
• m5.16xlarge
• m6i.xlarge
• c4.2xlarge
• c4.4xlarge
• c4.8xlarge
• r4.large
• r4.xlarge
• r4.2xlarge
• r4.4xlarge
• r4.8xlarge
• r4.16xlarge

**IMPORTANT**

If m4 instance types are not available in your region, such as with eu-west-3, use m5 types instead.

2. Copy the template from the CloudFormation template for worker machines section of this topic and save it as a YAML file on your computer. This template describes the networking objects and load balancers that your cluster requires.

3. If you specified an m5 instance type as the value for **WorkerInstanceType**, add that instance type to the **WorkerInstanceType.AllowedValues** parameter in the CloudFormation template.

4. Launch the CloudFormation template to create a stack of AWS resources that represent a worker node:

```
$ aws cloudformation create-stack --stack-name <name> 1
   --template-body file://<template>.yaml \ 2
   --parameters file://<parameters>.json 3
```

   1. `<name>` is the name for the CloudFormation stack, such as `cluster-worker-1`. You need the name of this stack if you remove the cluster.

   2. `<template>` is the relative path to and name of the CloudFormation template YAML file that you saved.
<parameters> is the relative path to and name of the CloudFormation parameters JSON file.

Example output

```
```

**NOTE**

The CloudFormation template creates a stack that represents one worker node.

5. Confirm that the template components exist:

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6. Continue to create worker stacks until you have created enough worker machines for your cluster. You can create additional worker stacks by referencing the same template and parameter files and specifying a different stack name.

**IMPORTANT**

You must create at least two worker machines, so you must create at least two stacks that use this CloudFormation template.

### 2.11.14.1. CloudFormation template for worker machines

You can use the following CloudFormation template to deploy the worker machines that you need for your OpenShift Container Platform cluster.

**Example 2.48. CloudFormation template for worker machines**

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
InfrastructureName:
  AllowedPattern: `^[a-zA-Z][a-zA-Z0-9-]{0,26}$`
  MaxLength: 27
  MinLength: 1
  ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
  Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
  Type: String
RhcosAmi:
  Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
  Type: AWS::EC2::Image::Id
Subnet:
  Description: The subnets, recommend private, to launch the master nodes into.
  Type: AWS::EC2::Subnet::Id
WorkerSecurityGroupId:
  Description: The master security group ID to associate with master nodes.
```
Type: AWS::EC2::SecurityGroup::Id
IgnitionLocation:
  Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
  Description: Ignition config file location.
  Type: String
CertificateAuthorities:
  Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
  Description: Base64 encoded certificate authority string to use.
  Type: String
WorkerInstanceProfileName:
  Description: IAM profile to associate with master nodes.
  Type: String
WorkerInstanceType:
  Default: m5.large
  Type: String
  AllowedValues:
  - "m4.large"
  - "m4.xlarge"
  - "m4.2xlarge"
  - "m4.4xlarge"
  - "m4.10xlarge"
  - "m4.16xlarge"
  - "m5.large"
  - "m5.xlarge"
  - "m5.2xlarge"
  - "m5.4xlarge"
  - "m5.8xlarge"
  - "m5.12xlarge"
  - "m5.16xlarge"
  - "m5a.large"
  - "m5a.xlarge"
  - "m5a.2xlarge"
  - "m5a.4xlarge"
  - "m5a.8xlarge"
  - "m5a.10xlarge"
  - "m5a.16xlarge"
  - "c4.large"
  - "c4.xlarge"
  - "c4.2xlarge"
  - "c4.4xlarge"
  - "c4.8xlarge"
  - "c5.large"
  - "c5.xlarge"
  - "c5.2xlarge"
  - "c5.4xlarge"
  - "c5.9xlarge"
  - "c5.12xlarge"
  - "c5.18xlarge"
  - "c5.24xlarge"
  - "c5a.large"
  - "c5a.xlarge"
  - "c5a.2xlarge"
  - "c5a.4xlarge"
  - "c5a.8xlarge"
  - "c5a.12xlarge"
  - "c5a.16xlarge"
- "c5a.24xlarge"
- "r4.large"
- "r4.xlarge"
- "r4.2xlarge"
- "r4.4xlarge"
- "r4.8xlarge"
- "r4.16xlarge"
- "r5.large"
- "r5.xlarge"
- "r5.2xlarge"
- "r5.4xlarge"
- "r5.8xlarge"
- "r5.12xlarge"
- "r5.16xlarge"
- "r5.24xlarge"
- "r5a.large"
- "r5a.xlarge"
- "r5a.2xlarge"
- "r5a.4xlarge"
- "r5a.8xlarge"
- "r5a.12xlarge"
- "r5a.16xlarge"
- "r5a.24xlarge"
- "t3.large"
- "t3.xlarge"
- "t3.2xlarge"
- "t3a.large"
- "t3a.xlarge"
- "t3a.2xlarge"

Metadata:
AWS::CloudFormation::Interface:
ParameterGroups:
- Label: "default: "Cluster Information"
  Parameters:
  - InfrastructureName
- Label: "default: "Host Information"
  Parameters:
  - WorkerInstanceType
  - RhcosAmi
  - IgnitionLocation
  - CertificateAuthorities
  - WorkerSecurityGroupId
  - WorkerInstanceProfileName
- Label: "default: "Network Configuration"
  Parameters:
  - Subnet
ParameterLabels:
Subnet:
  default: "Subnet"
InfrastructureName:
  default: "Infrastructure Name"
WorkerInstanceType:
2.11.15. Initializing the bootstrap sequence on AWS with user-provisioned infrastructure

After you create all of the required infrastructure in Amazon Web Services (AWS), you can start the bootstrap sequence that initializes the OpenShift Container Platform control plane.

Prerequisites
• You configured an AWS account.
• You added your AWS keys and region to your local AWS profile by running `aws configure`.
• You generated the Ignition config files for your cluster.
• You created and configured a VPC and associated subnets in AWS.
• You created and configured DNS, load balancers, and listeners in AWS.
• You created the security groups and roles required for your cluster in AWS.
• You created the bootstrap machine.
• You created the control plane machines.
• You created the worker nodes.

Procedure

1. Change to the directory that contains the installation program and start the bootstrap process that initializes the OpenShift Container Platform control plane:

   ```bash
   $ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \
   --log-level=info
   ```

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

INFO Waiting up to 20m0s for the Kubernetes API at https://api.mycluster.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s

If the command exits without a FATAL warning, your OpenShift Container Platform control plane has initialized.

NOTE

After the control plane initializes, it sets up the compute nodes and installs additional services in the form of Operators.

Additional resources

• See Monitoring installation progress for details about monitoring the installation, bootstrap, and control plane logs as an OpenShift Container Platform installation progresses.
2.11.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

2.11.17. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   ```

   Example output
The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.
NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the \texttt{oc exec}, \texttt{oc rsh}, and \texttt{oc logs} commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the \texttt{node-bootstrapper} service account in the \texttt{system:node} or \texttt{system:admin} groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

\begin{verbatim}
$ oc adm certificate approve <csr_name>
\end{verbatim}

\texttt{<csr_name>} is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

\begin{verbatim}
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
\end{verbatim}

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

\begin{verbatim}
$ oc get csr
\end{verbatim}

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the \texttt{Pending} status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

\begin{verbatim}
$ oc adm certificate approve <csr_name>
\end{verbatim}

\texttt{<csr_name>} is the name of a CSR from the list of current CSRs.
To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

**Example output**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

### 2.11.18. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

```
$ watch -n5 oc get clusteroperators
```

**Example output**

```
NAME               VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication     4.6.0  True  False  False  3h56m
cloud-credential   4.6.0  True  False  False  29h
cluster-autoscaler 4.6.0  True  False  False  29h
config-operator    4.6.0  True  False  False  6h39m
console            4.6.0  True  False  False  3h59m
```
2. Configure the Operators that are not available.

2.11.18.1. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```
  $ oc patch OperatorHub cluster --type json 
  -p '[["op": "add", "path": "/spec/disableAllDefaultSources", "value": true]]'
  ```

TIP

Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

2.11.18.2. Image registry storage configuration

Amazon Web Services provides default storage, which means the Image Registry Operator is available after installation. However, if the Registry Operator cannot create an S3 bucket and automatically configure storage, you must manually configure registry storage.
Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

2.11.18.2.1. Configuring registry storage for AWS with user-provisioned infrastructure

During installation, your cloud credentials are sufficient to create an Amazon S3 bucket and the Registry Operator will automatically configure storage.

If the Registry Operator cannot create an S3 bucket and automatically configure storage, you can create an S3 bucket and configure storage with the following procedure.

**Prerequisites**

- You have a cluster on AWS with user-provisioned infrastructure.
- For Amazon S3 storage, the secret is expected to contain two keys:
  - `REGISTRY_STORAGE_S3_ACCESSKEY`
  - `REGISTRY_STORAGE_S3_SECRETKEY`

**Procedure**

Use the following procedure if the Registry Operator cannot create an S3 bucket and automatically configure storage.

1. Set up a [Bucket Lifecycle Policy](#) to abort incomplete multipart uploads that are one day old.

2. Fill in the storage configuration in `configs.imageregistry.operator.openshift.io/cluster`:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io/cluster
   
   storage:
   s3:
   bucket: <bucket-name>
   region: <region-name>
   ```

**Example configuration**

- To secure your registry images in AWS, block public access to the S3 bucket.

2.11.18.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.
2.11.19. Deleting the bootstrap resources

After you complete the initial Operator configuration for the cluster, remove the bootstrap resources from Amazon Web Services (AWS).

**Prerequisites**

- You completed the initial Operator configuration for your cluster.

**Procedure**

1. Delete the bootstrap resources. If you used the CloudFormation template, delete its stack:
   - Delete the stack by using the AWS CLI:
     ```bash
     $ aws cloudformation delete-stack --stack-name <name>  
     
     <name> is the name of your bootstrap stack.
     ```
   - Delete the stack by using the AWS CloudFormation console.

2.11.20. Creating the Ingress DNS Records

If you removed the DNS Zone configuration, manually create DNS records that point to the Ingress load balancer. You can create either a wildcard record or specific records. While the following procedure uses A records, you can use other record types that you require, such as CNAME or alias.

**Prerequisites**
You deployed an OpenShift Container Platform cluster on Amazon Web Services (AWS) that uses infrastructure that you provisioned.

You installed the OpenShift CLI (oc).

You installed the jq package.

You downloaded the AWS CLI and installed it on your computer. See Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix).

**Procedure**

1. Determine the routes to create.

   - To create a wildcard record, use `*.apps.<cluster_name>.<domain_name>`, where `<cluster_name>` is your cluster name, and `<domain_name>` is the Route 53 base domain for your OpenShift Container Platform cluster.

   - To create specific records, you must create a record for each route that your cluster uses, as shown in the output of the following command:

     ```bash
     $ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}{end}' routes
     ```

     **Example output**

     ```
     oauth-openshift.apps.<cluster_name>.<domain_name>
     console-openshift-console.apps.<cluster_name>.<domain_name>
     downloads-openshift-console.apps.<cluster_name>.<domain_name>
     alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
     grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
     prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
     ```

2. Retrieve the Ingress Operator load balancer status and note the value of the external IP address that it uses, which is shown in the **EXTERNAL-IP** column:

   ```bash
   $ oc -n openshift-ingress get service router-default
   ```

   **Example output**

   ```
   NAME             TYPE           CLUSTER-IP      EXTERNAL-IP                            PORT(S)     AGE
   router-default   LoadBalancer   172.30.62.215   ab3...28.us-east-2.elb.amazonaws.com  80:31499/TCP,443:30693/TCP   5m
   ```

3. Locate the hosted zone ID for the load balancer:

   ```bash
   $ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName == "<external_ip>").CanonicalHostedZoneNameID'
   ```

   **For `<external_ip>`**, specify the value of the external IP address of the Ingress Operator load balancer that you obtained.
Example output

Z3AADJGX6KTTL2

The output of this command is the load balancer hosted zone ID.

4. Obtain the public hosted zone ID for your cluster’s domain:

```bash
$ aws route53 list-hosted-zones-by-name \
   --dns-name "<domain_name>" \
   --query 'HostedZones[? Config.PrivateZone != `true` && Name == `<domain_name>`.].Id' \
   --output text
```

1 2 For `<domain_name>`, specify the Route 53 base domain for your OpenShift Container Platform cluster.

Example output

/hostedzone/Z3URY6TWQ91KVV

The public hosted zone ID for your domain is shown in the command output. In this example, it is Z3URY6TWQ91KVV.

5. Add the alias records to your private zone:

```bash
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" -- 
   change-batch '{
   "Changes": [
   { 
   "Action": "CREATE",
   "ResourceRecordSet": {
   "Name": "\052.apps.<cluster_domain>",
   "Type": "A",
   "AliasTarget":{
   "HostedZoneId": "<hosted_zone_id>",
   "DNSName": "<external_ip>",
   "EvaluateTargetHealth": false
   }
   }
   }
   ]
   }
```

1 For `<private_hosted_zone_id>`, specify the value from the output of the CloudFormation template for DNS and load balancing.

2 For `<cluster_domain>`, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.

3 For `<hosted_zone_id>`, specify the public hosted zone ID for the load balancer that you obtained.
For `<external_ip>`, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

6. Add the records to your public zone:

```bash
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --change-batch '{
  "Changes": [
    {
      "Action": "CREATE",
      "ResourceRecordSet": {
        "Name": "\052.apps.<cluster_domain>",
        "Type": "A",
        "AliasTarget": {
          "HostedZoneId": "<hosted_zone_id>",
          "DNSName": "<external_ip>",
          "EvaluateTargetHealth": false
        }
      }
    }
  ]
}
```

1. For `<public_hosted_zone_id>`, specify the public hosted zone for your domain.
2. For `<cluster_domain>`, specify the domain or subdomain that you use with your OpenShift Container Platform cluster.
3. For `<hosted_zone_id>`, specify the public hosted zone ID for the load balancer that you obtained.
4. For `<external_ip>`, specify the value of the external IP address of the Ingress Operator load balancer. Ensure that you include the trailing period (.) in this parameter value.

2.11.21. Completing an AWS installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Amazon Web Service (AWS) user-provisioned infrastructure, monitor the deployment to completion.

**Prerequisites**

- You removed the bootstrap node for an OpenShift Container Platform cluster on user-provisioned AWS infrastructure.
- You installed the `oc` CLI.

**Procedure**

1. From the directory that contains the installation program, complete the cluster installation:

```bash
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

1
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Fe5en-ymBEc-Wt6NL"
INFO Time elapsed: 1s
```

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for **Recovering from expired control plane certificates** for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Register your cluster on the [Cluster registration](#) page.

### 2.11.22. Logging in to the cluster by using the web console

The **kubeadmin** user exists by default after an OpenShift Container Platform installation. You can log in to your cluster as the **kubeadmin** user by using the OpenShift Container Platform web console.

**Prerequisites**

- You have access to the installation host.
- You completed a cluster installation and all cluster Operators are available.

**Procedure**

1. Obtain the password for the **kubeadmin** user from the **kubeadmin-password** file on the installation host:

```
$ cat <installation_directory>/auth/kubeadmin-password
```
NOTE

Alternatively, you can obtain the kubeadmin password from the <installation_directory>/openshift_install.log log file on the installation host.

2. List the OpenShift Container Platform web console route:

```bash
$ oc get routes -n openshift-console | grep 'console-openshift'
```

NOTE

Alternatively, you can obtain the OpenShift Container Platform route from the <installation_directory>/openshift_install.log log file on the installation host.

Example output

```
console console-openshift-console.apps.<cluster_name>.<base_domain> console
https reencrypt/Redirect None
```

3. Navigate to the route detailed in the output of the preceding command in a web browser and log in as the kubeadmin user.

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

2.11.23. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

2.11.24. Additional resources

- See Working with stacks in the AWS documentation for more information about AWS CloudFormation stacks.

2.11.25. Next steps

- Validate an installation.
- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the must-gather tool.
• Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.

• If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.

• If necessary, you can opt out of remote health reporting.

• If necessary, you can remove cloud provider credentials.

2.12. UNINSTALLING A CLUSTER ON AWS

You can remove a cluster that you deployed to Amazon Web Services (AWS).

2.12.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

**NOTE**

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

• Have a copy of the installation program that you used to deploy the cluster.

• Have the files that the installation program generated when you created your cluster.

Procedure

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

   ```
   $ ./openshift-install destroy cluster \
   --dir <installation_directory> --log-level info
   ```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.
CHAPTER 3. INSTALLING ON AZURE

3.1. CONFIGURING AN AZURE ACCOUNT

Before you can install OpenShift Container Platform, you must configure a Microsoft Azure account.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

3.1.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default Azure subscription and service limits, quotas, and constraints affect your ability to install OpenShift Container Platform clusters.

**IMPORTANT**

Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
</table>

OpenShift Container Platform 4.6 Installing
A default cluster requires 40 vCPUs, so you must increase the account limit.

By default, each cluster creates the following instances:

- One bootstrap machine, which is removed after installation
- Three control plane machines
- Three compute machines

Because the bootstrap machine uses **Standard_D4s_v3** machines, which use 4 vCPUs, the control plane machines use **Standard_D8s_v3** virtual machines, which use 8 vCPUs, and the worker machines use **Standard_D4s_v3** virtual machines, which use 4 vCPUs, a default cluster requires 40 vCPUs. The bootstrap node VM, which uses 4 vCPUs, is used only during installation.

To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.

By default, the installation program distributes control plane and compute machines across all availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vCPU</td>
<td>40</td>
<td>20 per region</td>
<td>A default cluster requires 40 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances: - One bootstrap machine, which is removed after installation - Three control plane machines - Three compute machines Because the bootstrap machine uses <strong>Standard_D4s_v3</strong> machines, which use 4 vCPUs, the control plane machines use <strong>Standard_D8s_v3</strong> virtual machines, which use 8 vCPUs, and the worker machines use <strong>Standard_D4s_v3</strong> virtual machines, which use 4 vCPUs, a default cluster requires 40 vCPUs. The bootstrap node VM, which uses 4 vCPUs, is used only during installation. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require. By default, the installation program distributes control plane and compute machines across all availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.</td>
</tr>
<tr>
<td>Component</td>
<td>Number of components required by default</td>
<td>Default Azure limit</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OS Disk</td>
<td>7</td>
<td></td>
<td>VM OS disk must be able to sustain a minimum throughput of 5000 IOPS / 200MBps. This throughput can be provided by having a minimum of 1 TiB Premium SSD (P30). In Azure, disk performance is directly dependent on SSD disk sizes, so to achieve the throughput supported by <strong>Standard_D8s_v3</strong>, or other similar machine types available, and the target of 5000 IOPS, at least a P30 disk is required. Host caching must be set to <strong>ReadOnly</strong> for low read latency and high read IOPS and throughput. The reads performed from the cache, which is present either in the VM memory or in the local SSD disk, are much faster than the reads from the data disk, which is in the blob storage.</td>
</tr>
<tr>
<td>VNet</td>
<td>1</td>
<td>1000 per region</td>
<td>Each default cluster requires one Virtual Network (VNet), which contains two subnets.</td>
</tr>
<tr>
<td>Network interfaces</td>
<td>6</td>
<td>65,536 per region</td>
<td>Each default cluster requires six network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.</td>
</tr>
<tr>
<td>Network security groups</td>
<td>2</td>
<td>5000</td>
<td>Each default cluster Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>control plane</strong> Allows the control plane machines to be reached on port 6443 from anywhere</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>node</strong> Allows worker nodes to be reached from the Internet on ports 80 and 443</td>
</tr>
</tbody>
</table>
Each cluster creates the following load balancers:

**defaul t**  
Public IP address that load balances requests to ports 80 and 443 across worker machines

**internal**  
Private IP address that load balances requests to ports 6443 and 22623 across control plane machines

**external**  
Public IP address that load balances requests to port 6443 across control plane machines

If your applications create more Kubernetes `LoadBalancer` service objects, your cluster uses more load balancers.

<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network load balancers</td>
<td>3</td>
<td>1000 per region</td>
<td>Each cluster creates the following load balancers:</td>
</tr>
</tbody>
</table>
|                          |                                          |                    | **defaul t**  
Public IP address that load balances requests to ports 80 and 443 across worker machines

|                          |                                          |                    | **internal**  
Private IP address that load balances requests to ports 6443 and 22623 across control plane machines

|                          |                                          |                    | **external**  
Public IP address that load balances requests to port 6443 across control plane machines

| Public IP addresses      | 3                                        |                    | Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation. |
| Private IP addresses     | 7                                        |                    | The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address. |
| Spot VM vCPUs (optional) | 0                                        | 20 per region       | This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster. |

**NOTE**  
Using spot VMs for control plane nodes is not recommended.

### 3.1.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

**Procedure**
1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.

   **NOTE**

   For more information about purchasing domains through Azure, see Buy a custom domain name for Azure App Service in the Azure documentation.

2. If you are using an existing domain and registrar, migrate its DNS to Azure. See Migrate an active DNS name to Azure App Service in the Azure documentation.

3. Configure DNS for your domain. Follow the steps in the Tutorial: Host your domain in Azure DNS in the Azure documentation to create a public hosted zone for your domain or subdomain, extract the new authoritative name servers, and update the registrar records for the name servers that your domain uses.
   Use an appropriate root domain, such as openshiftcorp.com, or subdomain, such as clusters.openshiftcorp.com.

4. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain.

### 3.1.3. Increasing Azure account limits

To increase an account limit, file a support request on the Azure portal.

   **NOTE**

   You can increase only one type of quota per support request.

**Procedure**

1. From the Azure portal, click **Help + support** in the lower left corner.

2. Click **New support request** and then select the required values:
   a. From the **Issue type** list, select Service and subscription limits (quotas)
   b. From the **Subscription** list, select the subscription to modify.
   c. From the **Quota type** list, select the quota to increase. For example, select **Compute-VM (cores-vCPUs) subscription limit increases** to increase the number of vCPUs, which is required to install a cluster.
   d. Click **Next: Solutions**.

3. On the **Problem Details** page, provide the required information for your quota increase:
   a. Click **Provide details** and provide the required details in the **Quota details** window.
   b. In the SUPPORT METHOD and CONTACT INFO sections, provide the issue severity and your contact details.

4. Click **Next: Review + create** and then click **Create**.

### 3.1.4. Required Azure roles
OpenShift Container Platform needs a service principal so it can manage Microsoft Azure resources. Before you can create a service principal, your Azure account subscription must have the following roles:

- **User Access Administrator**
- **Owner**

To set roles on the Azure portal, see the Manage access to Azure resources using RBAC and the Azure portal in the Azure documentation.

### 3.1.5. Creating a service principal

Because OpenShift Container Platform and its installation program must create Microsoft Azure resources through Azure Resource Manager, you must create a service principal to represent it.

**Prerequisites**

- Install or update the **Azure CLI**.
- Install the **jq** package.
- Your Azure account has the required roles for the subscription that you use.

**Procedure**

1. Log in to the Azure CLI:

   ```
   $ az login
   Log in to Azure in the web console by using your credentials.
   ```

2. If your Azure account uses subscriptions, ensure that you are using the right subscription.
   a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

   ```
   $ az account list --refresh
   ```

   **Example output**

   ```
   [
   
   {  
   "cloudName": "AzureCloud",  
   "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",  
   "isDefault": true,  
   "name": "Subscription Name",  
   "state": "Enabled",  
   "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",  
   "user": {  
   "name": "you@example.com",  
   "type": "user"  
   }  
   }
   ]
   ```
b. View your active account details and confirm that the `tenantId` value matches the subscription you want to use:

```bash
$ az account show
```

**Example output**

```json
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

1. Ensure that the value of the `tenantId` parameter is the UUID of the correct subscription.

c. If you are not using the right subscription, change the active subscription:

```bash
$ az account set -s <id>
```

1. Substitute the value of the `id` for the subscription that you want to use for `<id>`.

d. If you changed the active subscription, display your account information again:

```bash
$ az account show
```

**Example output**

```json
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-de3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. Record the values of the `tenantId` and `id` parameters from the previous output. You need these values during OpenShift Container Platform installation.

4. Create the service principal for your account:
1. Replace `<service_principal>` with the name to assign to the service principal.

Example output

Changing "<service_principal>" to a valid URI of "http://<service_principal>", which is the required format used for service principal names

Retrying role assignment creation: 1/36
Retrying role assignment creation: 2/36
Retrying role assignment creation: 3/36
Retrying role assignment creation: 4/36

```
{
  "appId": "8bd0d04d-0ac2-43a8-928d-705c598c6956",
  "displayName": "<service_principal>",
  "name": "http://<service_principal>",
  "password": "ac461d78-bf4b-4387-ad16-7e32e328ae6",
  "tenant": "6048c7e9-b2ad-488d-a54e-dc3f6be6a7ee"
}
```

5. Record the values of the `appId` and `password` parameters from the previous output. You need these values during OpenShift Container Platform installation.

6. Grant additional permissions to the service principal.

   - You must always add the `Contributor` and `User Access Administrator` roles to the app registration service principal so the cluster can assign credentials for its components.

   - To operate the Cloud Credential Operator (CCO) in `mint mode`, the app registration service principal also requires the `Azure Active Directory Graph/Application.ReadWrite.OwnedBy` API permission.

   - To operate the CCO in `passthrough mode`, the app registration service principal does not require additional API permissions.

For more information about CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

a. To assign the `User Access Administrator` role, run the following command:

```
$ az role assignment create --role "User Access Administrator" \ 
  --assignee-object-id $(az ad sp list --filter "appId eq '<appId>'" \ 
  | jq '.[0].objectId' -r)  
```

Replace `<appld>` with the `appld` parameter value for your service principal.

b. To assign the `Azure Active Directory Graph` permission, run the following command:

```
$ az ad app permission add --id <appld> \ 
  --api 00000002-0000-0000-c000-000000000000 \ 
  --api-permissions 824c81eb-e3f8-4ee6-8f6d-de7f50d565b7=Role  
```

Replace `<appld>` with the `appld` parameter value for your service principal.
Example output

Invoking "az ad app permission grant --id 46d33abc-b8a3-46d8-8c84-f0fd58177435 --api 00000002-0000-0000-c000-000000000000" is needed to make the change effective.

For more information about the specific permissions that you grant with this command, see the GUID Table for Windows Azure Active Directory Permissions.

c. Approve the permissions request. If your account does not have the Azure Active Directory tenant administrator role, follow the guidelines for your organization to request that the tenant administrator approve your permissions request.

```
$ az ad app permission grant --id <appId> \
  --api 00000002-0000-0000-c000-000000000000
```

1 Replace `<appId>` with the `appId` parameter value for your service principal.

### 3.1.6. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription. The following Azure regions were tested and validated in OpenShift Container Platform version 4.6.1:

**Supported Azure public regions**

- `australiacentral` (Australia Central)
- `australiaeast` (Australia East)
- `australiasoutheast` (Australia South East)
- `brazilsouth` (Brazil South)
- `canadacentral` (Canada Central)
- `canadaeast` (Canada East)
- `centralindia` (Central India)
- `centralus` (Central US)
- `eastasia` (East Asia)
- `eastus` (East US)
- `eastus2` (East US 2)
- `francecentral` (France Central)
- `germanywestcentral` (Germany West Central)
- `japaneast` (Japan East)
- `japanwest` (Japan West)
• **koreacentral** (Korea Central)
• **koreasouth** (Korea South)
• **northcentralus** (North Central US)
• **northeurope** (North Europe)
• **norwayeast** (Norway East)
• **southafricanorth** (South Africa North)
• **southcentralus** (South Central US)
• **southeastasia** (Southeast Asia)
• **southindia** (South India)
• **switzerlandnorth** (Switzerland North)
• **uaenorth** (UAE North)
• **uksouth** (UK South)
• **ukwest** (UK West)
• **westcentralus** (West Central US)
• **westeurope** (West Europe)
• **westindia** (West India)
• **westus** (West US)
• **westus2** (West US 2)

**Supported Azure Government regions**
Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

• **usgovtexas** (US Gov Texas)
• **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the Azure documentation. Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

### 3.1.7. Next steps

- Install an OpenShift Container Platform cluster on Azure. You can **install a customized cluster** or **quickly install a cluster** with default options.

## 3.2. MANUALLY CREATING IAM FOR AZURE

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the
3.2.1. Alternatives to storing administrator-level secrets in the kube-system project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the credentialsMode parameter in the install-config.yaml file.

If you prefer not to store an administrator-level credential secret in the cluster kube-system project, you can set the credentialsMode parameter for the CCO to Manual when installing OpenShift Container Platform and manage your cloud credentials manually.

Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

Additional resources

- For a detailed description of all available CCO credential modes and their supported platforms, see the Cloud Credential Operator reference.

3.2.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster kube-system namespace.

Procedure

1. To generate the manifests, run the following command from the directory that contains the installation program:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

2. Insert a config map into the manifests directory so that the Cloud Credential Operator is placed in manual mode:

   ```
   $ cat <<EOF > mycluster/manifests/cco-configmap.yaml
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: cloud-credential-operator-config
     namespace: openshift-cloud-credential-operator
   annotations:
     release.openshift.io/create-only: "true"
   EOF
   ```
3. Remove the admin credential secret created using your local cloud credentials. This removal prevents your admin credential from being stored in the cluster:

```bash
$ rm mycluster/openshift/99_cloud-creds-secret.yaml
```

4. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your openshift-install binary is built to use:

```bash
$ openshift-install version
```

**Example output**

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

5. Locate all CredentialsRequest objects in this release image that target the cloud you are deploying on:

```bash
$ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --credentials-requests --cloud=azure
```

This displays the details for each request.

**Sample CredentialsRequest object**

```yaml
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
name: openshift-image-registry-azure
namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
      - role: Contributor
```

6. Create YAML files for secrets in the openshift-install manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the spec.secretRef for each credentialsRequest. The format for the secret data varies for each cloud provider.

7. From the directory that contains the installation program, proceed with your cluster creation:

```bash
$ openshift-install create cluster --dir <installation_directory>
```
IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state. For details, see the Upgrading clusters with manually maintained credentials section of the installation content for your cloud provider.

3.2.3. Admin credentials root secret format

Each cloud provider uses a credentials root secret in the kube-system namespace by convention, which is then used to satisfy all credentials requests and create their respective secrets. This is done either by minting new credentials, with mint mode, or by copying the credentials root secret, with passthrough mode.

The format for the secret varies by cloud, and is also used for each CredentialsRequest secret.

Microsoft Azure secret format

```yaml
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
  name: azure-credentials
stringData:
  azure_subscription_id: <SubscriptionID>
  azure_client_id: <ClientID>
  azure_client_secret: <ClientSecret>
  azure_tenant_id: <TenantID>
  azure_resource_prefix: <ResourcePrefix>
  azure_resourcegroup: <ResourceGroup>
  azure_region: <Region>

$ cat .openshift_install_state.json | jq "."installconfig.ClusterID".InfraID" -r

Example output

mycluster-2mpcn

This value would be used in the secret data as follows:

azure_resource_prefix: mycluster-2mpcn
azure_resourcegroup: mycluster-2mpcn-rg

3.2.4. Upgrading clusters with manually maintained credentials

If credentials are added in a future release, the Cloud Credential Operator (CCO) upgradable status for a cluster with manually maintained credentials changes to false. For minor release, for example, from 4.5 to 4.6, this status prevents you from upgrading until you have addressed any updated permissions.
For z-stream releases, for example, from 4.5.10 to 4.5.11, the upgrade is not blocked, but the credentials must still be updated for the new release.

Use the Administrator perspective of the web console to determine if the CCO is upgradeable.

1. Navigate to Administration → Cluster Settings.
2. To view the CCO status details, click cloud-credential in the Cluster Operators list.
3. If the Upgradeable status in the Conditions section is False, examine the credentialsRequests for the new release and update the manually maintained credentials on your cluster to match before upgrading.

In addition to creating new credentials for the release image that you are upgrading to, you must review the required permissions for existing credentials and accommodate any new permissions requirements for existing components in the new release. The CCO cannot detect these mismatches and will not set upgradable to false in this case.

The Manually creating IAM section of the installation content for your cloud provider explains how to obtain and use the credentials required for your cloud.

3.2.5. Mint mode

Mint mode is the default and recommended Cloud Credential Operator (CCO) credentials mode for OpenShift Container Platform. In this mode, the CCO uses the provided administrator-level cloud credential to run the cluster. Mint mode is supported for AWS, GCP, and Azure.

In mint mode, the admin credential is stored in the kube-system namespace and then used by the CCO to process the CredentialsRequest objects in the cluster and create users for each with specific permissions.

The benefits of mint mode include:

- Each cluster component has only the permissions it requires
- Automatic, on-going reconciliation for cloud credentials, including additional credentials or permissions that might be required for upgrades

One drawback is that mint mode requires admin credential storage in a cluster kube-system secret.

3.2.6. Next steps

- Install an OpenShift Container Platform cluster:
  - Installing a cluster quickly on Azure with default options on installer-provisioned infrastructure
  - Install a cluster with cloud customizations on installer-provisioned infrastructure
  - Install a cluster with network customizations on installer-provisioned infrastructure

3.3. INSTALLING A CLUSTER QUICKLY ON AZURE

In OpenShift Container Platform version 4.6, you can install a cluster on Microsoft Azure that uses the default configuration options.
3.3.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- **Configure an Azure account** to host the cluster and determine the tested and validated region to deploy the cluster to.
- If you use a firewall, you must **configure it to allow the sites** that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can **manually create and maintain IAM credentials**. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

3.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access **Quay.io** to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

3.3.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your **ssh-agent** and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user **core**. When you deploy the cluster, the key is added to the **core** user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as **AWS key pairs**.
Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ ssh-keygen -t ed25519 -N "\n-f <path>/<file_name> 1
```

Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

```bash
$ eval "$(ssh-agent -s)"
```

**Example output**

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```bash
$ ssh-add <path>/<file_name> 1
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.
3.3.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

3.3.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

   IMPORTANT

   You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites
Configure an account with the cloud platform that hosts your cluster.

Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   2. To view different installation details, specify warn, debug, or error instead of info.

   **IMPORTANT**

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Provide values at the prompts:

   a. Optional: Select an SSH key to use to access your cluster machines.

      **NOTE**

      For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

   b. Select azure as the platform to target.

   c. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

      - **azure subscription id** The subscription ID to use for the cluster. Specify the id value in your account output.

      - **azure tenant id** The tenant ID. Specify the tenantId value in your account output.

      - **azure service principal client id** The value of the appId parameter for the service principal.

      - **azure service principal client secret** The value of the password parameter for the service principal.

   d. Select the region to deploy the cluster to.
e. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

f. Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

g. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.
3.3.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.3.6.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   $ echo $PATH
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 3.3.6.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the `oc` binary to a directory that is on your `PATH`.

To check your `PATH`, open the command prompt and execute the following command:

```
C:\> path
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 3.3.6.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your `PATH`.

To check your `PATH`, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```
3.3.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadm credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami

   Example output
   
   system:admin

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.3.8. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

3.3.9. Next steps

- Customize your cluster.
If necessary, you can opt out of remote health reporting.

3.4. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a customized cluster on infrastructure that the installation program provisions on Microsoft Azure. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

3.4.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an Azure account to host the cluster and determine the tested and validated region to deploy the cluster to.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

3.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

3.4.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.
You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```
Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

3.4.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

3.4.5. Creating the installation configuration file
You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.
   
a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   Important: For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   **IMPORTANT**

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

   i. Optional: Select an SSH key to use to access your cluster machines.

   **NOTE**

   For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

   ii. Select **azure** as the platform to target.

   iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

   - **azure subscription id** The subscription ID to use for the cluster. Specify the `id` value in your account output.
   - **azure tenant id** The tenant ID. Specify the `tenantId` value in your account output.
   - **azure service principal client id** The value of the `appId` parameter for the service principal.
   - **azure service principal client secret** The value of the `password` parameter for the service principal.
iv. Select the region to deploy the cluster to.

v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

vi. Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 3.4.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

#### 3.4.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>

498
### Parameter  
**apiVersion**  
The API version for the `install-config.yaml` content. The current version is **v1**. The installer may also support older API versions.  

**baseDomain**  
The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the `baseDomain` and `metadata.name` parameter values that uses the `<metadata.name>.<baseDomain>` format.  

**metadata**  
Kubernetes resource `ObjectMeta`, from which only the `name` parameter is consumed.  

**metadata.name**  
The name of the cluster. DNS records for the cluster are all subdomains of `{.metadata.name}. {.baseDomain}`.  

**platform**  
The configuration for the specific platform upon which to perform the installation: `aws`, `baremetal`, `azure`, `openstack`, `ovirt`, `vsphere`. For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <strong>v1</strong>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{.metadata.name}. {.baseDomain}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
### Pull Secret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 3.4.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 3.2. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{ &quot;auths&quot;: { &quot;cloud.openshift.com&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; }, &quot;quay.io&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } } }</code></td>
</tr>
</tbody>
</table>

#### Networking

The configuration for the cluster network.

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.

<table>
<thead>
<tr>
<th>networking.network Type</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
| networking.clusterNetwork | The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap. | An array of objects. For example:
```
{ "networking": { "clusterNetwork": { "cidsr": 10.128.0.0/14, "hostPrefix": 23 } } }
``` |

Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
<td></td>
</tr>
</tbody>
</table>

### 3.4.5.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>serviceNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- cidr: 172.30.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>machineNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- cidr: 10.0.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>machineNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- cidr: 10.0.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>networking:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>machineNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- cidr: 10.0.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>networking:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>serviceNetwork:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- cidr: 172.30.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><strong>additionalTrustBundle</strong></td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td><strong>compute</strong></td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><strong>compute.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><strong>compute.hyperthreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td><strong>compute.name</strong></td>
<td>Required if you use <strong>compute</strong>. The name of the machine pool.</td>
<td><strong>worker</strong></td>
</tr>
<tr>
<td><strong>compute.platform</strong></td>
<td>Required if you use <strong>compute</strong>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <strong>controlPlane.platform</strong> parameter value.</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere</strong>, or <strong>{}</strong></td>
</tr>
<tr>
<td><strong>compute.replicas</strong></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to <strong>2</strong>. The default value is <strong>3</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td></td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hypertreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong> If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use <code>controlPlane</code>. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
### sshKey

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 3.4.5.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.platform.azure.osDisk.diskSizeGB</td>
<td>The Azure disk size for the VM.</td>
<td>Integer that represents the size of the disk in GB. The minimum supported disk size is 1024.</td>
</tr>
<tr>
<td>platform.azure.baseDomainResourceGroupName</td>
<td>The name of the resource group that contains the DNS zone for your base domain.</td>
<td>String, for example <code>production_cluster</code>.</td>
</tr>
<tr>
<td>platform.azure.outboundType</td>
<td>The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.</td>
<td><code>LoadBalancer</code> or <code>UserDefinedRouting</code>. The default is <code>LoadBalancer</code>.</td>
</tr>
<tr>
<td>platform.azure.region</td>
<td>The name of the Azure region that hosts your cluster.</td>
<td>Any valid region name, such as <code>centralus</code>.</td>
</tr>
<tr>
<td>platform.azure.zone</td>
<td>List of availability zones to place machines in. For high availability, specify at least two zones.</td>
<td>List of zones, for example <code>[&quot;1&quot;, &quot;2&quot;, &quot;3&quot;]</code>.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.azure.networkResourceGroupName</code></td>
<td>The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the <code>platform.azure.baseDomainResourceGroupName</code>.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.azure.virtualNetwork</code></td>
<td>The name of the existing VNet that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.azure.controlPlaneSubnet</code></td>
<td>The name of the existing subnet in your VNet that you want to deploy your control plane machines to.</td>
<td>Valid CIDR, for example <code>10.0.0.0/16</code>.</td>
</tr>
<tr>
<td><code>platform.azure.computeSubnet</code></td>
<td>The name of the existing subnet in your VNet that you want to deploy your compute machines to.</td>
<td>Valid CIDR, for example <code>10.0.0.0/16</code>.</td>
</tr>
<tr>
<td><code>platform.azure.cloudName</code></td>
<td>The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value <code>AzurePublicCloud</code> is used.</td>
<td>Any valid cloud environment, such as <code>AzurePublicCloud</code> or <code>AzureUSGovernmentCloud</code>.</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.

### 3.4.5.2. Sample customized `install-config.yaml` file for Azure

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com ①
controlPlane: ②
hyperthreading: Enabled ③④
name: master
platform:
```

①

②

③

④
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous
multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as Standard_D8s_v3, for your machines if you disable simultaneous multithreading.

You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes (also known as the master nodes) is 1024 GB.

Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

Specify the name of the resource group that contains the DNS zone for your base domain.

Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the `sshKey` value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 3.4.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>
     httpsProxy: https://<username>:<pswd>@<ip>:<port>
   noProxy: example.com
   additionalTrustBundle:
       ----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       ----END CERTIFICATE-----
   ...
   ```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   2. A proxy URL to use for creating HTTPS connections outside the cluster.

   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

   4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 3.4.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info
   ```

   **1** For `<installation_directory>`, specify the location of your customized `.install-config.yaml` file.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```bash
...  
INFO Install complete!  
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
```
NOTE

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3.4.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.4.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Linux Client** entry and save the file.

4. Unpack the archive:

   ```
   $ tar xzvf <file>
   ```

5. Place the `oc` binary in a directory that is on your **PATH**. To check your **PATH**, execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 3.4.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your **PATH**. To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 3.4.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

### 3.4.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   ```
   system:admin
   ```

**Additional resources**

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

### 3.4.9. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use **subscription watch** to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

### Additional resources
- See [About remote health monitoring](#) for more information about the Telemetry service

### 3.10. Next steps
- Customize your cluster.
- If necessary, you can [opt out of remote health reporting](#).

### 3.5. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Microsoft Azure. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

#### 3.5.1. Prerequisites
- Review details about the [OpenShift Container Platform installation and update](#) process.
- Configure an Azure account to host the cluster and determine the tested and validated region to deploy the cluster to.
- If you use a firewall, you must [configure it to allow the sites](#) that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can [manually create and maintain IAM credentials](#). Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

#### 3.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster. You must have Internet access to:
- Access [OpenShift Cluster Manager](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

3.5.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:
NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

    $ ssh-add <path>/<file_name> 1

**Example output**

    Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 3.5.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 3.5.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

#### Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

#### Procedure

1. Create the `install-config.yaml` file.
   a. Change to the directory that contains the installation program and run the following command:

      $ ./openshift-install create install-config --dir <installation_directory>  

      **FOR** `<installation_directory>`, **SPECIFY** the directory name to store the files that the installation program creates.

      **IMPORTANT**

      Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.
NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select azure as the platform to target.

iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

- **azure subscription id** The subscription ID to use for the cluster. Specify the `id` value in your account output.
- **azure tenant id** The tenant ID. Specify the `tenantId` value in your account output.
- **azure service principal client id** The value of the `appId` parameter for the service principal.
- **azure service principal client secret** The value of the `password` parameter for the service principal.

iv. Select the region to deploy the cluster to.

v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

vi. Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 3.5.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.
NOTE
After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT
The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

3.5.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 3.5. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>. <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}</code>. <code>{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere.**

For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform</strong></td>
<td>The configuration for the specific platform.</td>
<td><strong>Object</strong></td>
</tr>
<tr>
<td><strong>pullSecret</strong></td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auths&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;cloud.openshift.com&quot;: {</td>
</tr>
</tbody>
</table>
|             |                                                                             | "auth": "b3Blb=",
|             |                                                                             | "email": "you@example.com"   |
|             |                                                                             | },                            |
|             |                                                                             | "quay.io": {                 |
|             |                                                                             | "auth": "b3Blb=",
|             |                                                                             | "email": "you@example.com"   |
|             |                                                                             | }                            |
|             |                                                                             | }                            |

### 3.5.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 3.6. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td><strong>Object</strong></td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
### networking.network Type

The cluster network provider Container Network Interface (CNI) plug-in to install.

Either **OpenShiftSDN** or **OVNKubernetes**. The default value is **OpenShiftSDN**.

### networking.clusterNetwork

The IP address blocks for pods.

The default value is **10.128.0.0/14** with a host prefix of **/23**.

If you specify multiple IP address blocks, the blocks must not overlap.

### networking.clusterNetwork.cidr

Required if you use **networking.clusterNetwork**. An IP address block.

An IPv4 network.

### networking.clusterNetwork.hostPrefix

The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23** then each node is assigned a **/23** subnet out of the given **cidr**. A **hostPrefix** value of **23** provides **510** (2^(32 - 23) - 2) pod IP addresses.

### networking.serviceNetwork

The IP address block for services. The default value is **172.30.0.0/16**.

The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.

### networking.machineNetwork

The IP address blocks for machines.

If you specify multiple IP address blocks, the blocks must not overlap.

### Parameter | Description | Values
--- | --- | ---
**networking**: | Cluster network provider | Either **OpenShiftSDN** or **OVNKubernetes**. The default value is **OpenShiftSDN**.
| clusterNetwork | Container Network Interface (CNI) plug-in to install. | An array of objects. For example:
| cidr | The IP address blocks for pods. | An array of objects. For example:
| hostPrefix | The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23** then each node is assigned a **/23** subnet out of the given **cidr**. A **hostPrefix** value of **23** provides **510** (2^(32 - 23) - 2) pod IP addresses. | A subnet prefix.
| | | The default value is **23**.
| serviceNetwork | The IP address block for services. The default value is **172.30.0.0/16**. | An array with an IP address block in CIDR format. For example:
| machineNetwork | The IP address blocks for machines. | An array of objects. For example:
| | If you specify multiple IP address blocks, the blocks must not overlap. |
3.5.5.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 3.7. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td>NOTE</td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**
Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>
**fips**

Enable or disable FIPS mode. The default is **false** (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><strong>Internal</strong> or <strong>External</strong>. To deploy a private cluster, which cannot be accessed from the internet, set <strong>publish</strong> to <strong>Internal</strong>. The default value is <strong>External</strong>.</td>
</tr>
</tbody>
</table>
3.5.5.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table:

Table 3.8. Additional Azure parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.platform.azure.osDisk.diskSizeGB</td>
<td>The Azure disk size for the VM.</td>
<td>Integer that represents the size of the disk in GB. The minimum supported disk size is 1024.</td>
</tr>
<tr>
<td>platform.azure.baseDomainResourceGroupName</td>
<td>The name of the resource group that contains the DNS zone for your base domain.</td>
<td>String, for example production_cluster.</td>
</tr>
<tr>
<td>platform.azure.outboundType</td>
<td>The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.</td>
<td>LoadBalancer or UserDefinedRouting. The default is LoadBalancer.</td>
</tr>
<tr>
<td>platform.azure.region</td>
<td>The name of the Azure region that hosts your cluster.</td>
<td>Any valid region name, such as centralus.</td>
</tr>
<tr>
<td>platform.azure.zone</td>
<td>List of availability zones to place machines in. For high availability, specify at least two zones.</td>
<td>List of zones, for example [&quot;1&quot;, &quot;2&quot;, &quot;3&quot;]]</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>platform.azure.networkResourceGroupName</td>
<td>The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.virtualNetwork</td>
<td>The name of the existing VNet that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your control plane machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.computeSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your compute machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.cloudName</td>
<td>The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.</td>
<td>Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud.</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.

### 3.5.5.2. Sample customized install-config.yaml file for Azure

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane:
  hyperthreading: Enabled
  name: master
platform:
  azure:
```
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The `controlPlane` section is a single mapping, but the `compute` section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or `hyperthreading`. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to `Disabled`. If you disable simultaneous
multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as Standard_D8s_v3, for your machines if you disable simultaneous multithreading.

5. You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes (also known as the master nodes) is 1024 GB.

9. Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12. Specify the name of the resource group that contains the DNS zone for your base domain.

14. Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

16. Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

17. You can optionally provide the `sshKey` value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 3.5.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
  httpsProxy: https://<username>:<pswd>@<ip>:<port>
  noProxy: example.com
additionalTrustBundle:
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 3.5.6. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

**Phase 2**

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

### 3.5.7. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**
Create the `install-config.yaml` file and complete any modifications to it.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`
   
   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```bash
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```

   where:

   `<installation_directory>`
   
   Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

   **Specify a different VXLAN port for the OpenShift SDN network provider**

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     defaultNetwork:
       openshiftSDNConfig:
         vxlanPort: 4800
   ```

4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

### 3.5.8. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the Network API in the `operator.openshift.io` API group.
The CNO configuration inherits the following fields during cluster installation from the Network API in the Network.config.openshift.io API group and these fields cannot be changed:

**clusterNetwork**
- IP address pools from which pod IP addresses are allocated.

**serviceNetwork**
- IP address pool for services.

**defaultNetwork.type**
- Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the `defaultNetwork` object in the CNO object named `cluster`.

### 3.5.8.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always <code>cluster</code>.</td>
</tr>
<tr>
<td>spec.clusterNetwork</td>
<td>array</td>
<td>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.32.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the <code>install-config.yaml</code> file.</td>
</tr>
<tr>
<td>spec.serviceNetwork</td>
<td>array</td>
<td>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the <code>install-config.yaml</code> file.</td>
</tr>
<tr>
<td>spec.defaultNetwork</td>
<td>object</td>
<td>Configures the Container Network Interface (CNI) cluster network provider for the cluster network.</td>
</tr>
</tbody>
</table>
The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.

**defaultNetwork object configuration**
The values for the `defaultNetwork` object are defined in the following table:

### Table 3.10. defaultNetwork object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>type</strong></td>
<td>string</td>
<td>Either <a href="https://www.openshift.com">OpenShiftSDN</a> or <a href="https://www.ovn-kubernetes.io">OVNKubernetes</a>. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>openshiftSDNConfig</td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td>ovnKubernetesConfig</td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

**Configuration for the OpenShift SDN CNI cluster network provider**
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

### Table 3.11. openshiftSDNConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>mode</strong></td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is <strong>NetworkPolicy</strong>. The values <strong>Multitenant</strong> and <strong>Subnet</strong> are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>
The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.

This value cannot be changed after cluster installation.

The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.

If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.

On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.

**Example OpenShift SDN configuration**

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

**Configuration for the OVN-Kubernetes CNI cluster network provider**

The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

**Table 3.12. ovnKubernetesConfig object**
The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.

This value cannot be changed after cluster installation.

defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081

Example OVN-Kubernetes configuration

kubeProxyConfig object configuration
The values for the kubeProxyConfig object are defined in the following table:

Example OVN-Kubernetes configuration

defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081

kubeProxyConfig object configuration
The values for the kubeProxyConfig object are defined in the following table:

Table 3.13. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</td>
</tr>
</tbody>
</table>

NOTE
Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.
The minimum duration before refreshing `iptables` rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go `time` package. The default value is:

```
kubeProxyConfig:
  proxyArguments:
    iptables-min-sync-period:
      - 0s
```

### 3.5.9. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with OVN-Kubernetes. This allows a hybrid cluster that supports different node networking configurations. For example, this is necessary to run both Linux and Windows nodes in a cluster.

**IMPORTANT**

You must configure hybrid networking with OVN-Kubernetes during the installation of your cluster. You cannot switch to hybrid networking after the installation process.

**Prerequisites**

- You defined `OVN Kubernetes` for the `networking.networkType` parameter in the `install-config.yaml` file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`

   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: Network
   spec:
     EOF
   ```
<installation_directory>

Specifies the directory name that contains the manifests/ directory for your cluster.

3. Open the cluster-network-03-config.yml file in an editor and configure OVN-Kubernetes with hybrid networking, such as in the following example:

Specify a hybrid networking configuration

```yaml
apiVersion: operator.openshift.io/v1
type: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
  - cidr: 10.132.0.0/14
    hostPrefix: 23
    hybridOverlayVXLANPort: 9898 2
```

1 Specify the CIDR configuration used for nodes on the additional overlay network. The hybridClusterNetwork CIDR cannot overlap with the clusterNetwork CIDR.

2 Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default 4789 port. For more information on this requirement, see the Microsoft documentation on Pod-to-pod connectivity between hosts is broken.

4. Save the cluster-network-03-config.yml file and quit the text editor.

5. Optional: Back up the manifests/cluster-network-03-config.yml file. The installation program deletes the manifests/ directory when creating the cluster.

NOTE

For more information on using Linux and Windows nodes in the same cluster, see Understanding Windows container workloads.

3.5.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites
Configure an account with the cloud platform that hosts your cluster.

Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

### Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info
   ```

   

   1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

### Example output

```bash
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3.5.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.5.11.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ``

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 3.5.11.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your **PATH**.

   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 3.5.11.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your **PATH**.

   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```
3.5.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   Example output
   
   system:admin

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.5.13. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

3.5.14. Next steps

- Customize your cluster.
3.6. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET

In OpenShift Container Platform version 4.6, you can install a cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

3.6.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an Azure account to host the cluster and determine the tested and validated region to deploy the cluster to.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

3.6.2. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.6, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

3.6.2.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able
to access resources in both resource groups. For example, the Machine API controller attaches NICS for
the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet’s CIDR block must contain the **Networking.MachineCIDR** range, which is the IP
  address pool for cluster machines.

- The VNet and its subnets must belong to the same resource group, and the subnets must be
  configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the
compute machines. Because Azure distributes machines in different availability zones within the region
that you specify, your cluster will have high availability by default.

To ensure that the subnets that you provide are suitable, the installation program confirms the following
data:

- All the specified subnets exist.

- There are two private subnets, one for the control plane machines and one for the compute
  machines.

- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned
  in availability zones that you do not provide private subnets for. If required, the installation
  program creates public load balancers that manage the control plane and worker nodes, and
  Azure allocates a public IP address to them.

**NOTE**

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

### 3.6.2.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require
specific access to ensure that the cluster communication is correct. You must create rules to allow
access to the required cluster communication ports.

**IMPORTANT**

The network security group rules must be in place before you install the cluster. If you
attempt to install a cluster without the required access, the installation program cannot
reach the Azure APIs, and installation fails.

**Table 3.14. Required ports**

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Control plane</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Allows HTTP traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>443</td>
<td>Allows HTTPS traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>6443</td>
<td>Allows communication to the control plane machines</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
### 3.6.2.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

### 3.6.2.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

### 3.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access [Quay.io](#) to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

#### IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.
3.6.4. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```

   1. Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874
NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name>
```

Example output
```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

3.6.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

IMPORTANT
The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

IMPORTANT
Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

3.6.6. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.
   
   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   **IMPORTANT**

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

   i. Optional: Select an SSH key to use to access your cluster machines.

   **NOTE**

   For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

   ii. Select `azure` as the platform to target.
iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

- azure subscription id: The subscription ID to use for the cluster. Specify the id value in your account output.
- azure tenant id: The tenant ID. Specify the tenantId value in your account output.
- azure service principal client id: The value of the appId parameter for the service principal.
- azure service principal client secret: The value of the password parameter for the service principal.

iv. Select the region to deploy the cluster to.

v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

vi. Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 3.6.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.
IMPORTANT

The openshift-install command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

3.6.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 3.15. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;, &lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}, {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (−), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere**. For additional information about platform.<platform> parameters, consult the following table for your specific platform.

### pullSecret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 3.6.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 3.16. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the **networking** object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An IPv4 network.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
networking.machine Network.cidr

Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.

NOTE
Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

3.6.6.13. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 3.17. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following “Machine-pool” table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter Description Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>compute.hyperthreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><strong>compute.name</strong></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><strong>compute.platform</strong></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or {}</code></td>
</tr>
<tr>
<td><strong>compute.replicas</strong></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><strong>controlPlane</strong></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><strong>controlPlane.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Sources and repositories for the release–image content.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set publish to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
### sshKey

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 3.6.6.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
| sshKey    | The SSH key or keys to authenticate access your cluster machines. | One or more keys. For example: | sshKey:  
  <key1>  
  <key2>  
  <key3> |

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.azure.osDisk.diskSizeGB</td>
<td>The Azure disk size for the VM.</td>
<td>Integer that represents the size of the disk in GB. The minimum supported disk size is <strong>1024</strong>.</td>
</tr>
<tr>
<td>platform.azure.baseDomainResourceGroupName</td>
<td>The name of the resource group that contains the DNS zone for your base domain.</td>
<td>String, for example <strong>production_cluster</strong>.</td>
</tr>
<tr>
<td>platform.azure.outboundType</td>
<td>The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.</td>
<td><strong>LoadBalancer</strong> or <strong>UserDefinedRouting</strong>. The default is <strong>LoadBalancer</strong>.</td>
</tr>
<tr>
<td>platform.azure.region</td>
<td>The name of the Azure region that hosts your cluster.</td>
<td>Any valid region name, such as <strong>centralus</strong>.</td>
</tr>
<tr>
<td>platform.azure.zones</td>
<td>List of availability zones to place machines in. For high availability, specify at least two zones.</td>
<td>List of zones, for example <strong>[&quot;1&quot;, &quot;2&quot;, &quot;3&quot;]</strong>.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.azure.networkResourceGroupName</td>
<td>The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the <code>platform.azure.baseDomainResourceGroupName</code>.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.virtualNetwork</td>
<td>The name of the existing VNet that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your control plane machines to.</td>
<td>Valid CIDR, for example <code>10.0.0.0/16</code>.</td>
</tr>
<tr>
<td>platform.azure.computeSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your compute machines to.</td>
<td>Valid CIDR, for example <code>10.0.0.0/16</code>.</td>
</tr>
<tr>
<td>platform.azure.cloudName</td>
<td>The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value <code>AzurePublicCloud</code> is used.</td>
<td>Any valid cloud environment, such as <code>AzurePublicCloud</code> or <code>AzureUSGovernmentCloud</code>.</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.

### 3.6.6.2. Sample customized `install-config.yaml` file for Azure

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
hyperthreading: Enabled 3 4
name: master
platform:
  azure:
```
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes (also known as the master nodes) is 1024 GB.

Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

Specify the name of the resource group that contains the DNS zone for your base domain.

Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

If you use an existing VNet, specify the name of the resource group that contains it.

If you use an existing VNet, specify its name.

If you use an existing VNet, specify the name of the subnet to host the control plane machines.

If you use an existing VNet, specify the name of the subnet to host the compute machines.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

### 3.6.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.
Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

```
  NOTE
  The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.
```

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```
    apiVersion: v1
    baseDomain: my.domain.com
    proxy:
      httpProxy: http://<username>:<pswd>@<ip>:<port>  
      httpsProxy: https://<username>:<pswd>@<ip>:<port>  
    noProxy: example.com
    additionalTrustBundle: |
      -----BEGIN CERTIFICATE-----
      <MY_TRUSTED_CA_CERT>
      -----END CERTIFICATE-----
...  
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
4. If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the trustedCA field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

3.6.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info

   1

   For <installation_directory>, specify the location of your customized install-config.yaml file.

   2

   To view different installation details, specify warn, debug, or error instead of info.

   NOTE

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the kubeadmin user, display in your terminal.
Example output

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-WI5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

### 3.6.8. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of `oc`, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of `oc`.

#### 3.6.8.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.
Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:
   
   ```shell
   $ tar xvzf <file>
   ```

5. Place the `oc` binary in a directory that is on your PATH.
   
   To check your PATH, execute the following command:
   
   ```shell
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```shell
$ oc <command>
```

### 3.6.8.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH.
   
   To check your PATH, open the command prompt and execute the following command:
   
   ```cmd
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```cmd
C:\> oc <command>
```

### 3.6.8.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 macOS Client entry and save the file.

4. Unarchive the archive:
   
   ```shell
   $ tar xzf <file>
   ```

5. Place the `oc` binary in a directory that is on your PATH.
   
   To check your PATH, open the terminal and execute the following command:
   
   ```sh
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```sh
$ oc <command>
```

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

3.6.9. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.
3.6.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

3.6.11. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

3.7. INSTALLING A PRIVATE CLUSTER ON AZURE

In OpenShift Container Platform version 4.6, you can install a private cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

3.7.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an Azure account to host the cluster and determine the tested and validated region to deploy the cluster to.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

3.7.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the Internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

To deploy a private cluster, you must use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

Additionally, you must deploy a private cluster from a machine that has access the API services for the
cloud you provision to, the hosts on the network that you provision, and to the internet to obtain installation media. You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

3.7.2.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder in order to resolve the cluster’s private DNS records. The cluster’s machines use 168.63.129.16 internally for DNS resolution. For more information, see What is Azure Private DNS? and What is IP address 168.63.129.16? in the Azure documentation.

The cluster still requires access to Internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A BaseDomainResourceGroup, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

3.7.2.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

3.7.2.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the Internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the install-config.yaml file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the Internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.
You must ensure the following items are available before setting user-defined routing:

- Egress to the Internet is possible to pull container images, unless using an internal registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the Configuring your firewall section.

There are several pre-existing networking setups that are supported for Internet access using user-defined routing.

**Private cluster with network address translation**
You can use Azure VNET network address translation (NAT) to provide outbound Internet access for the subnets in your cluster. You can reference Create a NAT gateway using Azure CLI in the Azure documentation for configuration instructions.

When using a VNet setup with Azure NAT and user-defined routing configured, you can create a private cluster with no public endpoints.

**Private cluster with Azure Firewall**
You can use Azure Firewall to provide outbound routing for the VNet used to install the cluster. You can learn more about providing user-defined routing with Azure Firewall in the Azure documentation.

When using a VNet setup with Azure Firewall and user-defined routing configured, you can create a private cluster with no public endpoints.

**Private cluster with a proxy configuration**
You can use a proxy with user-defined routing to allow egress to the Internet. You must ensure that cluster Operators do not access Azure APIs using a proxy; Operators must have access to Azure APIs outside of the proxy.

When using the default route table for subnets, with 0.0.0.0/0 populated automatically by Azure, all Azure API requests are routed over Azure’s internal network even though the IP addresses are public. As long as the Network Security Group rules allow egress to Azure API endpoints, proxies with user-defined routing configured allow you to create private clusters with no public endpoints.

**Private cluster with no Internet access**
You can have VNets with no access to the Internet if your cluster has access to the following:

- An internal registry mirror that allows for pulling container images
- Access to Azure APIs

With these requirements available, you can use user-defined routing to create private clusters with no public endpoints.

### 3.7.3. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.6, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. This is a good option to use if you cannot obtain the infrastructure creation
permissions that are required to create the VNet.

### 3.7.3.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet’s CIDR block must contain the `Networking.MachineCIDR` range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for.

**NOTE**

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

### 3.7.3.1.1. Network security group requirements
The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

**IMPORTANT**

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.19. Required ports

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Control plane</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Allows HTTP traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>443</td>
<td>Allows HTTPS traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>6443</td>
<td>Allows communication to the control plane machines</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>22623</td>
<td>Allows communication to the machine config server</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

Since cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo–network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

3.7.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as V Nets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make V Nets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.7.3.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

3.7.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.
You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 3.7.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "\" \\
   -f <path>/<file_name> \\
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.
If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   $ eval "$(ssh-agent -s)"

   **Example output**

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   $ ssh-add <path>/<file_name>  

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

   **Next steps**

   - When you install OpenShift Container Platform, provide the SSH public key to the installation program.

3.7.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   
   $ mkdir <installation_directory>
   
   $ mkdir <installation_directory>
   
   $ tar xvf openshift-install-linux.tar.gz
   
   $ mkdir <installation_directory>
   ``

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 3.7.7. Manually creating the installation configuration file

For installations of a private OpenShift Container Platform cluster that are only accessible from an internal network and are not visible to the Internet, you must manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```bash
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**
   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**
   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 3.7.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

   **NOTE**
   After installation, you cannot modify these parameters in the `install-config.yaml` file.

   **IMPORTANT**
   The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

#### 3.7.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 3.20. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
| pullSecret | Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.                                                                                                                                                             | {  
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
3.7.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 3.21. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the networking object after installation.</td>
</tr>
<tr>
<td>networking.networkType</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^{(32 - 23) - 2}) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td>The default value is 23.</td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <strong>172.30.0.0/16</strong>.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking: serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking: machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: <strong>10.0.0.0/16</strong></td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The default value is <strong>10.0.0.0/16</strong> for all platforms other than libvirt. For libvirt, the default value is <strong>192.168.126.0/24</strong>.</td>
<td>An IP network block in CIDR notation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, <strong>10.0.0.0/16</strong></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
</tr>
</tbody>
</table>

### 3.7.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

#### Table 3.22. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>compute.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><code>Enabled</code> or <code>Disabled</code></td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td><code>worker</code></td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hypertreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use <strong>controlPlane</strong>. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use <strong>controlPlane</strong>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <strong>compute.platform</strong> parameter value.</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere, or {}</strong></td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.</td>
<td></td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</td>
</tr>
</tbody>
</table>
Required if you use `imageContentSources`. Specify the repository that users refer to, for example, in image pull specifications.

Specify one or more repositories that may also contain the same images.

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

**Values**

- **String**
- **Array of strings**
- **Internal** or **External**. To deploy a private cluster, which cannot be accessed from the internet, set `publish` to **Internal**. The default value is **External**.
- One or more keys. For example:

  ```
  sshKey:
  <key1>
  <key2>
  <key3>
  ```

3.7.7.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table:

**Table 3.23. Additional Azure parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>controlPlane.platform.azure.osDisk.diskSizeGB</code></td>
<td>The Azure disk size for the VM.</td>
<td>Integer that represents the size of the disk in GB. The minimum supported disk size is 1024.</td>
</tr>
<tr>
<td><code>platform.azure.baseDomainResourceGroupName</code></td>
<td>The name of the resource group that contains the DNS zone for your base domain.</td>
<td>String, for example <code>production_cluster</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>platform.azure.outboundType</td>
<td>The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.</td>
<td>LoadBalancer or UserDefinedRouting. The default is LoadBalancer.</td>
</tr>
<tr>
<td>platform.azure.region</td>
<td>The name of the Azure region that hosts your cluster.</td>
<td>Any valid region name, such as centralus.</td>
</tr>
<tr>
<td>platform.azure.zone</td>
<td>List of availability zones to place machines in. For high availability, specify at least two zones.</td>
<td>List of zones, for example [&quot;1&quot;, &quot;2&quot;, &quot;3&quot;].</td>
</tr>
<tr>
<td>platform.azure.networkResourceGroupName</td>
<td>The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.virtualNetwork</td>
<td>The name of the existing VNet that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your control plane machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.computeSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your compute machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.cloudName</td>
<td>The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.</td>
<td>Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud.</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.

3.7.7.2. Sample customized install-config.yaml file for Azure
You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024
        type: Standard_D8s_v3
      replicas: 3
  compute:
    - hyperthreading: Enabled
      name: worker
      platform:
        azure:
          type: Standard_D2s_v3
          osDisk:
            diskSizeGB: 512
          zones:
            - "1"
            - "2"
            - "3"
      replicas: 5
metadata:
  name: test-cluster
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  azure:
    baseDomainResourceGroupName: resource_group
    region: centralus
    resourceGroupName: existing_resource_group
    networkResourceGroupName: vnet_resource_group
    virtualNetwork: vnet
    controlPlaneSubnet: control_plane_subnet
    computeSubnet: compute_subnet
    outboundType: UserDefinedRouting
```
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as Standard_D8s_v3, for your machines if you disable simultaneous multithreading.

You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes (also known as the master nodes) is 1024 GB.

Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

Specify the name of the resource group that contains the DNS zone for your base domain.

Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

If you use an existing VNet, specify the name of the resource group that contains it.

If you use an existing VNet, specify its name.

If you use an existing VNet, specify the name of the subnet to host the control plane machines.

If you use an existing VNet, specify the name of the subnet to host the compute machines.

You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.
The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

How to publish the user-facing endpoints of your cluster. Set publish to Internal to deploy a private cluster, which cannot be accessed from the Internet. The default value is External.

### 3.7.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

**NOTE**

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```
1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 3.7.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
• Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info

   
   For `<installation_directory>`, specify the

   To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   
   NOTE

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

   Example output

   
   ...
   INFO Install complete!
   INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
   INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
   INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
   INFO Time elapsed: 36m22s

   
   NOTE

   The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3.7.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.7.9.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH
After you install the OpenShift CLI, it is available using the `oc` command:

```bash
$ oc <command>
```

### 3.7.9.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the `oc` binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   ```bash
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```bash
C:\> oc <command>
```

### 3.7.9.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```bash
$ oc <command>
```
3.7.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   `system:admin`

**Additional resources**

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

3.7.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service

3.7.12. Next steps

- Customize your cluster.
3.8. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION

In OpenShift Container Platform version 4.6, you can install a cluster on Microsoft Azure into a government region. To configure the government region, you modify parameters in the `install-config.yaml` file before you install the cluster.

3.8.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an Azure account to host the cluster and determine the tested and validated government region to deploy the cluster to.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

3.8.2. Azure government regions

OpenShift Container Platform supports deploying a cluster to Microsoft Azure Government (MAG) regions. MAG is specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads on Azure. MAG is composed of government-only data center regions, all granted an Impact Level 5 Provisional Authorization.

Installing to a MAG region requires manually configuring the Azure Government dedicated cloud instance and region in the `install-config.yaml` file. You must also update your service principal to reference the appropriate government environment.

**NOTE**

The Azure government region cannot be selected using the guided terminal prompts from the installation program. You must define the region manually in the `install-config.yaml` file. Remember to also set the dedicated cloud instance, like `AzureUSGovernmentCloud`, based on the region specified.

3.8.3. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the Internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

To deploy a private cluster, you must use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
Additionally, you must deploy a private cluster from a machine that has access the API services for the cloud you provision to, the hosts on the network that you provision, and to the internet to obtain installation media. You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

### 3.8.3.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder in order to resolve the cluster’s private DNS records. The cluster’s machines use **168.63.129.16** internally for DNS resolution. For more information, see [What is Azure Private DNS?](#) and [What is IP address 168.63.129.16?](#) in the Azure documentation.

The cluster still requires access to Internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A **BaseDomainResourceGroup**, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

### 3.8.3.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

### 3.8.3.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the Internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the `install-config.yaml` file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the Internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.
You must ensure the following items are available before setting user-defined routing:

- Egress to the Internet is possible to pull container images, unless using an internal registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the Configuring your firewall section.

There are several pre-existing networking setups that are supported for Internet access using user-defined routing.

**Private cluster with network address translation**
You can use Azure VNET network address translation (NAT) to provide outbound Internet access for the subnets in your cluster. You can reference Create a NAT gateway using Azure CLI in the Azure documentation for configuration instructions.

When using a VNet setup with Azure NAT and user-defined routing configured, you can create a private cluster with no public endpoints.

**Private cluster with Azure Firewall**
You can use Azure Firewall to provide outbound routing for the VNet used to install the cluster. You can learn more about providing user-defined routing with Azure Firewall in the Azure documentation.

When using a VNet setup with Azure Firewall and user-defined routing configured, you can create a private cluster with no public endpoints.

**Private cluster with a proxy configuration**
You can use a proxy with user-defined routing to allow egress to the Internet. You must ensure that cluster Operators do not access Azure APIs using a proxy; Operators must have access to Azure APIs outside of the proxy.

When using the default route table for subnets, with 0.0.0.0/0 populated automatically by Azure, all Azure API requests are routed over Azure’s internal network even though the IP addresses are public. As long as the Network Security Group rules allow egress to Azure API endpoints, proxies with user-defined routing configured allow you to create private clusters with no public endpoints.

**Private cluster with no Internet access**
You can have VNets with no access to the Internet if your cluster has access to the following:

- An internal registry mirror that allows for pulling container images
- Access to Azure APIs

With these requirements available, you can use user-defined routing to create private clusters with no public endpoints.

### 3.8.4. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.6, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. This is a good option to use if you cannot obtain the infrastructure creation
permissions that are required to create the VNet.

3.8.4.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICs for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet’s CIDR block must contain the `Networking.MachineCIDR` range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.

**NOTE**

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.
3.8.4.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

**IMPORTANT**

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.24. Required ports

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Control plane</th>
<th>Compute</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>Allows HTTP traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>443</td>
<td>Allows HTTPS traffic</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>6443</td>
<td>Allows communication to the control plane machines</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>22623</td>
<td>Allows communication to the machine config server</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

Since cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

3.8.4.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.8.4.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

3.8.5. Internet access for OpenShift Container Platform
In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

3.8.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.
Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the **ssh-agent** process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the **ssh-agent**:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   **Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa**

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**3.8.7. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   $ mkdir <installation_directory>
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 3.8.8. Manually creating the installation configuration file

When installing OpenShift Container Platform on Microsoft Azure into a government region, you must manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```bash
   $ mkdir <installation_directory>
   ```
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

**NOTE**

You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 3.8.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 3.8.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 3.25. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

3.8.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 3.26. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>

```yaml
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR notation). The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given cidr. A <code>hostPrefix</code> value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use <code>networking.machineNetwork</code>. An IP address block.</td>
<td>An IP network block in CIDR notation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, <strong>10.0.0.0/16</strong>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set the <code>networking.machineNetwork</code> to match the CIDR that the preferred NIC resides in.</td>
</tr>
</tbody>
</table>

3.8.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 3.27. Optional parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>additionalTrustBundle</strong></td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td><strong>compute</strong></td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><strong>compute.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><strong>compute.hyperthreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><strong>compute.name</strong></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><strong>compute.platform</strong></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{} </code></td>
</tr>
<tr>
<td><strong>compute.replicas</strong></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hypertexting, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td>IMPORTANT</td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.</td>
<td></td>
</tr>
<tr>
<td>NOTE</td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. To deploy a private cluster, which cannot be accessed from the internet, set publish to Internal. The default value is External.</td>
</tr>
</tbody>
</table>
The SSH key or keys to authenticate access your cluster machines.

**NOTE**
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</code></td>
</tr>
</tbody>
</table>

### 3.8.8.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table:

**Table 3.28. Additional Azure parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.platform.azure.osDisk.diskSizeGB</td>
<td>The Azure disk size for the VM.</td>
<td>Integer that represents the size of the disk in GB. The minimum supported disk size is 1024.</td>
</tr>
<tr>
<td>platform.azure.baseDomainResourceGroupName</td>
<td>The name of the resource group that contains the DNS zone for your base domain.</td>
<td>String, for example production_cluster.</td>
</tr>
<tr>
<td>platform.azure.outboundType</td>
<td>The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing.</td>
<td>LoadBalancer or UserDefinedRouting. The default is LoadBalancer.</td>
</tr>
<tr>
<td>platform.azure.region</td>
<td>The name of the Azure region that hosts your cluster.</td>
<td>Any valid region name, such as centralus.</td>
</tr>
<tr>
<td>platform.azure.zone</td>
<td>List of availability zones to place machines in. For high availability, specify at least two zones.</td>
<td>List of zones, for example [&quot;1&quot;, &quot;2&quot;, &quot;3&quot;].</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.azure.networkResourceGroupName</td>
<td>The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the platform.azure.baseDomainResourceGroupName.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.virtualNetwork</td>
<td>The name of the existing VNet that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td>platform.azure.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your control plane machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.computeSubnet</td>
<td>The name of the existing subnet in your VNet that you want to deploy your compute machines to.</td>
<td>Valid CIDR, for example 10.0.0.0/16.</td>
</tr>
<tr>
<td>platform.azure.cloudName</td>
<td>The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value AzurePublicCloud is used.</td>
<td>Any valid cloud environment, such as AzurePublicCloud or AzureUSGovernmentCloud.</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.

### 3.8.8.2. Sample customized install-config.yaml file for Azure

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your install-config.yaml file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com  
controlPlane:           
  hyperthreading: Enabled
name: master
platform:
```
Required.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift
Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as Standard_D8s_v3, for your machines if you disable simultaneous multithreading.

You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes (also known as the master nodes) is 1024 GB.

Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

Specify the name of the resource group that contains the DNS zone for your base domain.

Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

If you use an existing VNet, specify the name of the resource group that contains it.

If you use an existing VNet, specify its name.

If you use an existing VNet, specify the name of the subnet to host the control plane machines.

If you use an existing VNet, specify the name of the subnet to host the compute machines.

You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.

Specify the name of the Azure cloud environment to deploy your cluster to. Set AzureUSGovernmentCloud to deploy to a Microsoft Azure Government (MAG) region. The default value is AzurePublicCloud.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.
NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

How to publish the user-facing endpoints of your cluster. Set publish to Internal to deploy a private cluster, which cannot be accessed from the Internet. The default value is External.

3.8.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the install-config.yaml file.

Prerequisites

- You have an existing install-config.yaml file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE
The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE
Only the Proxy object named cluster is supported, and no additional proxies can be created.

3.8.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure
1. Change to the directory that contains the installation program and initialize the cluster deployment:
$ ./openshift-install create cluster --dir <installation_directory> \ 
  --log-level=info

1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
3.8.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.8.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

3.8.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
   
   ```
   C:> path
   ```

   After you install the OpenShift CLI, it is available using the oc command:
   
   ```
   C:> oc <command>
   ```

### 3.8.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   
   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:
   
   ```
   $ oc <command>
   ```

### 3.8.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadmin credentials:
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   $ oc whoami

   Example output

   system:admin

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

### 3.8.12. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use [subscription watch](#) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

### 3.8.13. Next steps

- Customize your cluster.

- If necessary, you can [opt out of remote health reporting](#).

### 3.9. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES

In OpenShift Container Platform version 4.6, you can install a cluster on Microsoft Azure by using infrastructure that you provide.

Several [Azure Resource Manager](#) (ARM) templates are provided to assist in completing these steps or to help model your own.
3.9.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure an Azure account to host the cluster.
- Download the Azure CLI and install it on your computer. See Install the Azure CLI in the Azure documentation. The documentation below was last tested using version 2.2.0 of the Azure CLI. Azure CLI commands might perform differently based on the version you use.
- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

3.9.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

3.9.3. Configuring your Azure project
Before you can install OpenShift Container Platform, you must configure an Azure project to host it.

### IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see [Resolve reserved resource name errors](#) in the Azure documentation.

#### 3.9.3.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default [Azure subscription and service limits, quotas, and constraints](#) affect your ability to install OpenShift Container Platform clusters.

### IMPORTANT

Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
</table>

---

621
<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
</table>
| vCPU      | 40                                       | 20 per region       | A default cluster requires 40 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances:  
  - One bootstrap machine, which is removed after installation  
  - Three control plane machines  
  - Three compute machines  
  Because the bootstrap machine uses **Standard_D4s_v3** machines, which use 4 vCPUs, the control plane machines use **Standard_D8s_v3** virtual machines, which use 8 vCPUs, and the worker machines use **Standard_D4s_v3** virtual machines, which use 4 vCPUs, a default cluster requires 40 vCPUs. The bootstrap node VM, which uses 4 vCPUs, is used only during installation.  
  To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.  
  By default, the installation program distributes control plane and compute machines across all availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones. |

---

622
<table>
<thead>
<tr>
<th>Component</th>
<th>Number of components required by default</th>
<th>Default Azure limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS Disk</td>
<td>7</td>
<td></td>
<td>VM OS disk must be able to sustain a minimum throughput of 5000 IOPS / 200MBps. This throughput can be provided by having a minimum of 1 TiB Premium SSD (P30). In Azure, disk performance is directly dependent on SSD disk sizes, so to achieve the throughput supported by <strong>Standard_D8s_v3</strong>, or other similar machine types available, and the target of 5000 IOPS, at least a P30 disk is required. Host caching must be set to <strong>ReadOnly</strong> for low read latency and high read IOPS and throughput. The reads performed from the cache, which is present either in the VM memory or in the local SSD disk, are much faster than the reads from the data disk, which is in the blob storage.</td>
</tr>
<tr>
<td>VNet</td>
<td>1</td>
<td>1000 per region</td>
<td>Each default cluster requires one Virtual Network (VNet), which contains two subnets.</td>
</tr>
<tr>
<td>Network interfaces</td>
<td>6</td>
<td>65,536 per region</td>
<td>Each default cluster requires six network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.</td>
</tr>
<tr>
<td>Network security groups</td>
<td>2</td>
<td>5000</td>
<td>Each default cluster Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>control plane</strong> Allows the control plane machines to be reached on port 6443 from anywhere</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>node</strong> Allows worker nodes to be reached from the Internet on ports 80 and 443</td>
</tr>
</tbody>
</table>
Each cluster creates the following load balancers:

- **default** (Public IP address) that load balances requests to ports 80 and 443 across worker machines.
- **internal** (Private IP address) that load balances requests to ports 6443 and 22623 across control plane machines.
- **external** (Public IP address) that load balances requests to port 6443 across control plane machines.

If your applications create more Kubernetes LoadBalancer service objects, your cluster uses more load balancers.

- **Public IP addresses**: Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.

- **Private IP addresses**: The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.

- **Spot VM vCPUs (optional)**: If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node. This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster.

### 3.9.3.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

#### Procedure
1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.

   **NOTE**
   For more information about purchasing domains through Azure, see Buy a custom domain name for Azure App Service in the Azure documentation.

2. If you are using an existing domain and registrar, migrate its DNS to Azure. See Migrate an active DNS name to Azure App Service in the Azure documentation.

3. Configure DNS for your domain. Follow the steps in the Tutorial: Host your domain in Azure DNS in the Azure documentation to create a public hosted zone for your domain or subdomain, extract the new authoritative name servers, and update the registrar records for the name servers that your domain uses. Use an appropriate root domain, such as openshiftcorp.com, or subdomain, such as clusters.openshiftcorp.com.

4. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain.

You can view Azure’s DNS solution by visiting this example for creating DNS zones.

### 3.9.3.3. Increasing Azure account limits

To increase an account limit, file a support request on the Azure portal.

   **NOTE**
   You can increase only one type of quota per support request.

**Procedure**

1. From the Azure portal, click Help + support in the lower left corner.

2. Click New support request and then select the required values:
   a. From the Issue type list, select Service and subscription limits (quotas)
   b. From the Subscription list, select the subscription to modify.
   c. From the Quota type list, select the quota to increase. For example, select Compute-VM (cores-vCPUs) subscription limit increases to increase the number of vCPUs, which is required to install a cluster.
   d. Click Next: Solutions.

3. On the Problem Details page, provide the required information for your quota increase:
   a. Click Provide details and provide the required details in the Quota details window.
   b. In the SUPPORT METHOD and CONTACT INFO sections, provide the issue severity and your contact details.

4. Click Next: Review + create and then click Create.
3.9.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

3.9.3.5. Required Azure roles

OpenShift Container Platform needs a service principal so it can manage Microsoft Azure resources. Before you can create a service principal, your Azure account subscription must have the following roles:

- **User Access Administrator**
- **Owner**

To set roles on the Azure portal, see the Manage access to Azure resources using RBAC and the Azure portal in the Azure documentation.

3.9.3.6. Creating a service principal

Because OpenShift Container Platform and its installation program must create Microsoft Azure resources through Azure Resource Manager, you must create a service principal to represent it.

**Prerequisites**

- Install or update the Azure CLI.
- Install the `jq` package.
- Your Azure account has the required roles for the subscription that you use.

**Procedure**

1. Log in to the Azure CLI:

   ```
   $ az login
   
   Log in to Azure in the web console by using your credentials.
   ```

2. If your Azure account uses subscriptions, ensure that you are using the right subscription.

   a. View the list of available accounts and record the `tenantId` value for the subscription you want to use for your cluster:

   ```
   $ az account list --refresh
   
   Example output
   ```

   ```
   [  
     {  
       "cloudName": "AzureCloud",
   ```
b. View your active account details and confirm that the `tenantId` value matches the subscription you want to use:

```
$ az account show
```

**Example output**

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

Ensure that the value of the `tenantId` parameter is the UUID of the correct subscription.

c. If you are not using the right subscription, change the active subscription:

```
$ az account set -s <id> 1
```

1 Substitute the value of the `id` for the subscription that you want to use for `<id>`.

d. If you changed the active subscription, display your account information again:

```
$ az account show
```

**Example output**

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
```
3. Record the values of the **tenantId** and **id** parameters from the previous output. You need these values during OpenShift Container Platform installation.

4. Create the service principal for your account:

   ```bash
   $ az ad sp create-for-rbac --role Contributor --name <service_principal>  
   
   1 Replace `<service_principal>` with the name to assign to the service principal.
   
   **Example output**

   Changing "<service_principal>" to a valid URI of "http://<service_principal>", which is the required format used for service principal names
   Retrying role assignment creation: 1/36
   Retrying role assignment creation: 2/36
   Retrying role assignment creation: 3/36
   Retrying role assignment creation: 4/36
   
   {  
     "appId": "8bd0d04d-0ac2-43a8-928d-705c598c6956",
     "displayName": "<service_principal>",
     "name": "http://<service_principal>",
     "password": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
     "tenant": "6048c7e9-b2ad-488d-a54e-dc3f6be6a7ee"
   }
   
   1
   
   5. Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

6. Grant additional permissions to the service principal.

   - You must always add the **Contributor** and **User Access Administrator** roles to the app registration service principal so the cluster can assign credentials for its components.
   - To operate the Cloud Credential Operator (CCO) in **mint mode**, the app registration service principal also requires the **Azure Active Directory Graph/Application.ReadWrite.OwnedBy** API permission.
   - To operate the CCO in **passthrough mode**, the app registration service principal does not require additional API permissions.

   For more information about CCO modes, see the **Cloud Credential Operator** entry in the **Red Hat Operators reference** content.

   a. To assign the **User Access Administrator** role, run the following command:
b. To assign the Azure Active Directory Graph permission, run the following command:

```bash
$ az role assignment create --role "User Access Administrator" \
--assignee-object-id $(az ad sp list --filter "appId eq '<appId>'" \
| jq '.[0].objectId' -r)
```

Replace `<appId>` with the `appId` parameter value for your service principal.

```
Replace <appId> with the appId parameter value for your service principal.
```

Example output

Invoking "az ad app permission grant --id 46d33abc-b8a3-46d8-8c84-f0fd58177435 --api 00000002-0000-0000-c000-000000000000 --api-permissions 824c81eb-e3f8-4ee6-8f6d-de7f50d565b7=Role

For more information about the specific permissions that you grant with this command, see the GUID Table for Windows Azure Active Directory Permissions.

c. Approve the permissions request. If your account does not have the Azure Active Directory tenant administrator role, follow the guidelines for your organization to request that the tenant administrator approve your permissions request.

```
$ az ad app permission grant --id <appId>
```

Replace `<appId>` with the `appId` parameter value for your service principal.

3.9.3.7. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription. The following Azure regions were tested and validated in OpenShift Container Platform version 4.6.1:

**Supported Azure public regions**

- **australiacentral** (Australia Central)
- **australiaseast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaelst** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **southafricannorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **switzerlandnorth** (Switzerland North)
- **uabenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)

**Supported Azure Government regions**
Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
You can reference all available MAG regions in the Azure documentation. Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

### 3.9.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 3.9.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.
NOTE
In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N " " \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874

   NOTE
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```
   $ ssh-add <path>/<file_name>
   ```
Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1. Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

3.9.6. Creating the installation files for Azure

To install OpenShift Container Platform on Microsoft Azure using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the `install-config.yaml` file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate `var` partition during the preparation phases of installation.

3.9.6.1. Optional: Creating a separate `/var` partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

**IMPORTANT**

If you follow the steps to create a separate `/var` partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure
1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

**Example output**

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and $HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the `clusterconfig/openshift` directory:

```
$ ls $HOME/clusterconfig/openshift/
```

**Example output**

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...```

4. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  
          partitions:
            - label: var
              startMiB: <partition_start_offset>  
              sizeMiB: <partition_size>
          filesystems:
1. The storage device name of the disk that you want to partition.

2. When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

3. The size of the data partition in mebibytes.

4. The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.

5. The prjquota mount option must be enabled for filesystems used for container storage.

NOTE
When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

```bash
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

3.9.6.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.
Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   IMPORTANT
   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

      NOTE
      For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

      ii. Select `azure` as the platform to target.

      iii. If you do not have a Microsoft Azure profile stored on your computer, specify the following Azure parameter values for your subscription and service principal:

          - `azure subscription id` The subscription ID to use for the cluster. Specify the `id` value in your account output.

          - `azure tenant id` The tenant ID. Specify the `tenantId` value in your account output.

          - `azure service principal client id` The value of the `appId` parameter for the service principal.

          - `azure service principal client secret` The value of the `password` parameter for the service principal.

      iv. Select the region to deploy the cluster to.
v. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

vi. Enter a descriptive name for your cluster.

**IMPORTANT**

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

vii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

c. Optional: If you do not want the cluster to provision compute machines, empty the compute pool by editing the resulting `install-config.yaml` file to set `replicas` to 0 for the compute pool:

```yaml
compute:
  - hyperthreading: Enabled
    name: worker
    platform: {}
    replicas: 0
```

Set to 0.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

3.9.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 3.9.6.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure.

**NOTE**

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Export common variables found in the `install-config.yaml` to be used by the provided ARM templates:

   ```bash
   $ export CLUSTER_NAME=<cluster_name>
   $ export AZURE_REGION=<azure_region>
   $ export SSH_KEY=<ssh_key>
   $ export BASE_DOMAIN=<base_domain>
   $ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group>
   
   1. The value of the `.metadata.name` attribute from the `install-config.yaml` file.
   2. The region to deploy the cluster into, for example `centralus`. This is the value of the `.platform.azure.region` attribute from the `install-config.yaml` file.
   3. The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the `.sshKey` attribute from the `install-config.yaml` file.
   4. The base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster. This is the value of the `.baseDomain` attribute from the `install-config.yaml` file.
   5. The resource group where the public DNS zone exists. This is the value of the `.platform.azure.baseDomainResourceGroupName` attribute from the `install-config.yaml` file.

For example:

```bash
$ export CLUSTER_NAME=<cluster_name>
$ export AZURE_REGION=<azure_region>
$ export SSH_KEY=<ssh_key>
$ export BASE_DOMAIN=<base_domain>
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group>
```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster

2. Export the kubeadmin credentials:

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

3.9.6.5. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.

- You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

$ ./openshift-install create manifests --dir <installation_directory>

Example output
1. For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
   
   By removing these files, you prevent the cluster from automatically generating control plane machines.
   
   3. Remove the Kubernetes manifest files that define the worker machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   
   Because you create and manage the worker machines yourself, you do not need to initialize these machines.
   
   4. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

   5. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

   ```yaml
   apiVersion: config.openshift.io/v1
   kind: DNS
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     baseDomain: example.openshift.com
     privateZone:
       id: mycluster-100419-private-zone
     publicZone:
       id: example.openshift.com
   status: {}
   
   1 2 Remove this section completely.

   If you do so, you must add ingress DNS records manually in a later step.
6. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

a. Export the infrastructure ID by using the following command:

```
$ export INFRA_ID=<infra_id>
```

The OpenShift Container Platform cluster has been assigned an identifier (INFRA_ID) in the form of `<cluster_name>-<random_string>`. This will be used as the base name for most resources created using the provided ARM templates. This is the value of the .status.infrastructureName attribute from the manifests/cluster-infrastructure-02-config.yml file.

b. Export the resource group by using the following command:

```
$ export RESOURCE_GROUP=<resource_group>
```

All resources created in this Azure deployment exists as part of a resource group. The resource group name is also based on the INFRA_ID, in the form of `<cluster_name>-<random_string>-rg`. This is the value of the .status.platformStatus.azure.resourceGroupName attribute from the manifests/cluster-status-02-config.yml file.

7. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

- auth
  - kubeadmin-password
  - kubeconfig
- bootstrap.ign
- master.ign
- metadata.json
- worker.ign

3.9.7. Creating the Azure resource group and identity

You must create a Microsoft Azure resource group and an identity for that resource group. These are both used during the installation of your OpenShift Container Platform cluster on Azure.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
Procedure

1. Create the resource group in a supported Azure region:

   $ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}

2. Create an Azure identity for the resource group:

   $ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity

   This is used to grant the required access to Operators in your cluster. For example, this allows
   the Ingress Operator to create a public IP and its load balancer. You must assign the Azure
   identity to a role.

3. Grant the Contributor role to the Azure identity:

   a. Export the following variables required by the Azure role assignment:

      $ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`

      $ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`

   b. Assign the Contributor role to the identity:

      $ az role assignment create --assignee "$PRINCIPAL_ID" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"

3.9.8. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally; therefore, you must copy
and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a
storage container so they are accessible during deployment.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

   $ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
**WARNING**

The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your `CLUSTER_NAME` variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see [Resolve errors for storage account names](#) in the Azure documentation.

2. Export the storage account key as an environment variable:

   ```bash
   $ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --account-name ${CLUSTER_NAME}sa --query '[0].value' -o tsv`
   ```

3. Choose the RHCOS version to use and export the URL of its VHD to an environment variable:

   ```bash
   $ export VHD_URL=`curl -s https://raw.githubusercontent.com/openshift/installer/release-4.6/data/data/rhcos.json | jq -r .azure.url`
   ```

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Copy the chosen VHD to a blob:

   ```bash
   $ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
   $ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri "$VHD_URL"
   ```

To track the progress of the VHD copy task, run this script:

```bash
status=unknown
while [ "$status" != "success" ]
do
   status=`az storage blob show --container-name vhd --name "rhcos.vhd" --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv --query properties.copy.status`
echo $status
done
```

5. Create a blob storage container and upload the generated `bootstrap.ign` file:
3.9.9. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, Azure's DNS solution is used, so you will create a new public DNS zone for external (internet) visibility and a private DNS zone for internal cluster resolution.

**NOTE**

The public DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the public DNS zone; be sure the installation config you generated earlier reflects that scenario.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

**Procedure**

1. Create the new public DNS zone in the resource group exported in the `BASE_DOMAIN_RESOURCE_GROUP` environment variable:

   ```bash
   $ az network dns zone create -g $(BASE_DOMAIN_RESOURCE_GROUP) -n $(CLUSTER_NAME).$(BASE_DOMAIN)
   ```

   You can skip this step if you are using a public DNS zone that already exists.

2. Create the private DNS zone in the same resource group as the rest of this deployment:

   ```bash
   $ az network private-dns zone create -g $(RESOURCE_GROUP) -n $(CLUSTER_NAME).$(BASE_DOMAIN)
   ```

   You can learn more about configuring a public DNS zone in Azure by visiting that section.

3.9.10. Creating a VNet in Azure

You must create a virtual network (VNet) in Microsoft Azure for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.
NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the ARM template for the VNet section of this topic and save it as 01_vnet.json in your cluster’s installation directory. This template describes the VNet that your cluster requires.

2. Create the deployment by using the az CLI:

   ```
   $ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/01_vnet.json" \
   --parameters baseName="${INFRA_ID}"  
   
   1 The base name to be used in resource names; this is usually the cluster’s infrastructure ID.
   ```

3. Link the VNet template to the private DNS zone:

   ```
   $ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z 
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v 
   "$\{INFRA\_ID\}\_vnet" -e false
   ```

3.9.10.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

```
Example 3.1. 01_vnet.json ARM template

{
    "contentVersion" : "1.0.0.0",
    "parameters" : {
        "baseName" : {
            "type" : "string",
            "minLength" : 1,
            "metadata" : {
                "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
            }
        }
    }
}
```


"variables": {
      "location": "[resourceGroup().location]",
      "virtualNetworkName": "[concat(parameters('baseName'), '-vnet')]",
      "addressPrefix": "10.0.0.0/16",
      "masterSubnetName": "[concat(parameters('baseName'), '-master-subnet')]",
      "masterSubnetPrefix": "10.0.0.0/24",
      "nodeSubnetName": "[concat(parameters('baseName'), '-worker-subnet')]",
      "nodeSubnetPrefix": "10.0.1.0/24",
      "clusterNsgName": "[concat(parameters('baseName'), '-nsg')]"
    },
    "resources": [
      {
        "apiVersion": "2018-12-01",
        "type": "Microsoft.Network/virtualNetworks",
        "name": "[variables('virtualNetworkName')]",
        "location": "[variables('location')]",
        "dependsOn": [
          "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
        ],
        "properties": {
          "addressSpace": {
            "addressPrefixes": [
              "[variables('addressPrefix')]"
            ]
          },
          "subnets": [
            {
              "name": "[variables('masterSubnetName')]",
              "properties": {
                "addressPrefix": "[variables('masterSubnetPrefix')]",
                "serviceEndpoints": [],
                "networkSecurityGroup": {
                  "id": "[resourceId('Microsoft.Network/networkSecurityGroups',
                                      variables('clusterNsgName'))]"
                }
              }
            },
            {
              "name": "[variables('nodeSubnetName')]",
              "properties": {
                "addressPrefix": "[variables('nodeSubnetPrefix')]",
                "serviceEndpoints": [],
                "networkSecurityGroup": {
                  "id": "[resourceId('Microsoft.Network/networkSecurityGroups',
                                      variables('clusterNsgName'))]"
                }
              }
            }
          ]
        }
      },
      {
        "type": "Microsoft.Network/networkSecurityGroups",
        "name": "[variables('clusterNsgName')]",
        "apiVersion": "2018-10-01",
        "location": "[variables('location')]"
      }
    ]
3.9.11. Deploying the RHCOS cluster image for the Azure infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure for your OpenShift Container Platform nodes.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.
- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the ARM template for image storage section of this topic and save it as `02_storage.json` in your cluster’s installation directory. This template describes the image storage that your cluster requires.

2. Export the RHCOS VHD blob URL as a variable:

   ```bash
   $ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
   ```

3. Deploy the cluster image:

   ```bash
   $ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/02_storage.json" \
   --parameters vhdBlobURL="${VHD_BLOB_URL}"  \
   --parameters baseName="${INFRA_ID}"  \
   ```
The blob URL of the RHCOS VHD to be used to create master and worker machines.

The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3.9.11.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

Example 3.2. 02_storage.json ARM template

```json
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and worker machines is located"
      }
    }
  },
  "variables": {
    "location": [resourceGroup().location],
    "imageName": [concat(parameters('baseName'), '-image')]
  },
  "resources": [
    {
      "apiVersion": "2018-06-01",
      "type": "Microsoft.Compute/images",
      "name": [variables('imageName')],
      "location": [variables('location')],
      "properties": {
        "storageProfile": {
          "osDisk": {
            "osType": "Linux",
            "osState": "Generalized",
            "blobUri": [parameters('vhdBlobURL')],
            "storageAccountType": "Standard_LRS"
          }
        }
      }
    }
  ]
}
```
3.9.12. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config from the machine config server.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 3.29. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td>9000-9999</td>
<td></td>
<td>Host level services, including the node exporter on ports <strong>9100-9101</strong> and the Cluster Version Operator on port <strong>9099</strong>.</td>
</tr>
<tr>
<td>10250-10259</td>
<td></td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td>10256</td>
<td></td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td>6081</td>
<td></td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td>9000-9999</td>
<td></td>
<td>Host level services, including the node exporter on ports <strong>9100-9101</strong>.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 3.30. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 3.31. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements

The infrastructure that you provision for your cluster must meet the following network topology requirements.
IMPORTANT

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   IMPORTANT
   
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 3.32. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.
2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

**Table 3.33. Application Ingress load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

### 3.9.13. Creating networking and load balancing components in Azure

You must configure networking and load balancing in Microsoft Azure for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.

**NOTE**

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
Procedure

1. Copy the template from the **ARM template for the network and load balancers** section of this topic and save it as `03_infra.json` in your cluster’s installation directory. This template describes the networking and load balancing objects that your cluster requires.

2. Create the deployment by using the `az` CLI:

   ```bash
   $ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/03_infra.json" \
   --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \
   --parameters baseName="${INFRA_ID}"
   
   1 The name of the private DNS zone.
   2 The base name to be used in resource names; this is usually the cluster’s infrastructure ID.
   ```

3. Create an **API DNS record** in the public zone for the API public load balancer. The `$[BASE_DOMAIN_RESOURCE_GROUP]` variable must point to the resource group where the public DNS zone exists.

   a. Export the following variable:

   ```bash
   $ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
   
   b. Create the DNS record in a new public zone:

   ```bash
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
   
   c. If you are adding the cluster to an existing public zone, you can create the DNS record in it instead:

   ```bash
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n api.$[CLUSTER_NAME] -a ${PUBLIC_IP} --ttl 60
   ```

3.9.13.1. **ARM template for the network and load balancers**

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

**Example 3.3. 03_infra.json ARM template**

```json
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
```
"variables": {
  "location": [resourceGroup().location],
  "virtualNetworkName": [concat(parameters('baseName'), '-vnet')],
  "virtualNetworkID": [resourceId('Microsoft.Network/virtualNetworks', variables('virtualNetworkName'))],
  "masterSubnetName": [concat(parameters('baseName'), '-master-subnet')],
  "masterSubnetRef": [concat(variables('virtualNetworkID'), '/subnets/', variables('masterSubnetName'))],
  "masterPublicIpAddressName": [concat(parameters('baseName'), '-master-pip')],
  "masterPublicIpAddressID": [resourceId('Microsoft.Network/publicIPAddresses', variables('masterPublicIpAddressName'))],
  "masterLoadBalancerName": [concat(parameters('baseName'), '-public-lb')],
  "masterLoadBalancerID": [resourceId('Microsoft.Network/loadBalancers', variables('masterLoadBalancerName'))],
  "internalLoadBalancerName": [concat(parameters('baseName'), '-internal-lb')],
  "internalLoadBalancerID": [resourceId('Microsoft.Network/loadBalancers', variables('internalLoadBalancerName'))],
  "skuName": "Standard"
},
"resources": [
  {
    "apiVersion": "2018-12-01",
    "type": "Microsoft.Network/publicIPAddresses",
    "name": [variables('masterPublicIpAddressName')],
    "location": [variables('location')],
    "sku": {
      "name": [variables('skuName')]
    },
    "properties": {
      "publicIPAddressAllocationMethod": "Static",
      "dnsSettings": {
        "domainNameLabel": [variables('masterPublicIpAddressName')]
      }
    }
  },
  {
    "apiVersion": "2018-12-01",
    "type": "Microsoft.Network/loadBalancers",
    "name": [variables('masterLoadBalancerName')],
    "location": [variables('location')],
    "sku": {
      "name": [variables('skuName')]
    },
    "dependsOn": [
      [concat('Microsoft.Network/publicIPAddresses', variables('masterPublicIpAddressName'))]
    ]
  }
]
"properties": {
  "frontendIPConfigurations": [
    {
      "name": "public-lb-ip",
      "properties": {
        "publicIPAddress": {
          "id": "[variables('masterPublicIpAddressID')]
        }
      }
    }
  ],
  "backendAddressPools": [
    {
      "name": "public-lb-backend"
    }
  ],
  "loadBalancingRules": [
    {
      "name": "api-internal",
      "properties": {
        "frontendIPConfiguration": {
          "id": "[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-ip')]"
        },
        "backendAddressPool": {
          "id": "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/public-lb-backend')]"
        },
        "protocol": "Tcp",
        "loadDistribution": "Default",
        "idleTimeoutInMinutes": 30,
        "frontendPort": 6443,
        "backendPort": 6443,
        "probe": {
          "id": "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
        }
      }
    }
  ],
  "probes": [
    {
      "name": "api-internal-probe",
      "properties": {
        "protocol": "Https",
        "port": 6443,
        "requestPath": "/readyz",
        "intervalInSeconds": 10,
        "numberOfProbes": 3
      }
    }
  ]
},

"apiVersion": "2018-12-01",
"type": "Microsoft.Network/loadBalancers", 
"name": "[variables('internalLoadBalancerName')]",
"location": "[variables('location')]",
"sku": {
  "name": "[variables('skuName')]"
},
"properties": {
  "frontendIPConfigurations": [
    {
      "name": "internal-lb-ip",
      "properties": {
        "privateIPAllocationMethod": "Dynamic",
        "subnet": {
          "id": "[variables('masterSubnetRef')]"
        },
        "privateIPAddressVersion": "IPv4"
      }
    }
  ],
  "backendAddressPools": [
    {
      "name": "internal-lb-backend"
    }
  ],
  "loadBalancingRules": [
    {
      "name": "api-internal",
      "properties": {
        "frontendIPConfiguration": {
          "id": "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-ip')]"
        },
        "frontendPort": 6443,
        "backendPort": 6443,
        "enableFloatingIP": false,
        "idleTimeoutInMinutes": 30,
        "protocol": "Tcp",
        "enableTcpReset": false,
        "loadDistribution": "Default",
        "backendAddressPool": {
          "id": "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
        },
        "probe": {
          "id": "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
        }
      }
    }
  ],
  "name": "sint",
  "properties": {
    "frontendIPConfiguration": {
      "id": "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-ip')]"
    }
  }
},
"frontendPort": 6443,
"backendPort": 6443,
"enableFloatingIP": false,
"idleTimeoutInMinutes": 30,
"protocol": "Tcp",
"enableTcpReset": false,
"loadDistribution": "Default",
"backendAddressPool": {
  "id": "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
},
"probe": {
  "id": "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
}
}
"enableFloatingIP" : false,
"idleTimeoutInMinutes" : 30,
"protocol" : "Tcp",
"enableTcpReset" : false,
"loadDistribution" : "Default",
"backendAddressPool" : {
  "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]
},
"probe" : {
  "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]
}
],
"probes" : [
  {
    "name" : "api-internal-probe",
    "properties" : {
      "protocol" : "Https",
      "port" : 6443,
      "requestPath" : "/readyz",
      "intervalInSeconds" : 10,
      "numberOfProbes" : 3
    }
  },
  {
    "name" : "sint-probe",
    "properties" : {
      "protocol" : "Https",
      "port" : 22623,
      "requestPath" : "/healthz",
      "intervalInSeconds" : 10,
      "numberOfProbes" : 3
    }
  }
],
"apiVersion" : "2018-09-01",
"type" : "Microsoft.Network/privateDnsZones/A",
"name" : "/subscriptions/0b1f6470-1f78-450c-ba28-cdd6b73e7b2a/resourceGroups/my-resource-group/providers/Microsoft.Network/privateDnsZones/A[concat(parameters('privateDNSZoneName'), '/api')]
",
"location" : "/subscriptions/0b1f6470-1f78-450c-ba28-cdd6b73e7b2a/resourceGroups/my-resource-group/providers/Microsoft.Location/locations",
"dependsOn" : [
  
  
],
"properties" : {
  "ttl" : 60,
  "aRecords" : [
    {
      "ipv4Address" : 
      [reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIPAddress]
    }
  ]

You must create the bootstrap machine in Microsoft Azure to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.

Procedure

1. Copy the template from the ARM template for the bootstrap machine section of this topic and save it as `04_bootstrap.json` in your cluster’s installation directory. This template describes the bootstrap machine that your cluster requires.
2. Export the following variables required by the bootstrap machine deployment:

```bash
$ export BOOTSTRAP_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c "files" -n "bootstrap.ign" -o tsv`
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.1.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d "n"
```

3. Create the deployment by using the `az` CLI:

```bash
$ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/04_bootstrap.json" \
   --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1   --parameters sshKeyData="${SSH_KEY}" \ 2   --parameters baseName="${INFRA_ID}" \ 3
```

1. The bootstrap Ignition content for the bootstrap cluster.
2. The SSH RSA public key file as a string.
3. The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

### 3.9.14.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

**Example 3.4. 04_bootstrap.json ARM template**

```json
{
   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
   "contentVersion": "1.0.0.0",
   "parameters": {
      "baseName": {
         "type": "string",
         "minLength": 1,
         "metadata": {
            "description": "Base name to be used in resource names (usually the cluster’s Infra ID)"
         }
      },
      "bootstrapIgnition": {
         "type": "string",
         "minLength": 1,
         "metadata": {
            "description": "Bootstrap ignition content for the bootstrap cluster"
         }
      },
      "sshKeyData": {
         "type": "securestring",
         "metadata": {
            "description": "SSH RSA public key file as a string."
         }
      }
   }
}
```
"bootstrapVMSize": {
  "type": "string",
  "defaultValue": "Standard_D4s_v3",
  "allowedValues": [
    "Standard_A2",
    "Standard_A3",
    "Standard_A4",
    "Standard_A5",
    "Standard_A6",
    "Standard_A7",
    "Standard_A8",
    "Standard_A9",
    "Standard_A10",
    "Standard_A11",
    "Standard_D2",
    "Standard_D3",
    "Standard_D4",
    "Standard_D11",
    "Standard_D12",
    "Standard_D13",
    "Standard_D14",
    "Standard_D2_v2",
    "Standard_D3_v2",
    "Standard_D4_v2",
    "Standard_D5_v2",
    "Standard_D8_v3",
    "Standard_D11_v2",
    "Standard_D12_v2",
    "Standard_D13_v2",
    "Standard_D14_v2",
    "Standard_E2_v3",
    "Standard_E4_v3",
    "Standard_E8_v3",
    "Standard_E16_v3",
    "Standard_E32_v3",
    "Standard_E64_v3",
    "Standard_E2s_v3",
    "Standard_E4s_v3",
    "Standard_E8s_v3",
    "Standard_E16s_v3",
    "Standard_E32s_v3",
    "Standard_E64s_v3",
    "Standard_G1",
    "Standard_G2",
    "Standard_G3",
    "Standard_G4",
    "Standard_G5",
    "Standard_DS2",
    "Standard_DS3",
    "Standard_DS4",
    "Standard_DS11",
    "Standard_DS12",
    "Standard_DS13",
    "Standard_DS14",
    "Standard_DS2_v2",
    "Standard_DS3_v2"
  ]
}
"Standard_DS4_v2",
"Standard_DS5_v2",
"Standard_DS11_v2",
"Standard_DS12_v2",
"Standard_DS13_v2",
"Standard_DS14_v2",
"Standard_GS1",
"Standard_GS2",
"Standard_GS3",
"Standard_GS4",
"Standard_GS5",
"Standard_D2s_v3",
"Standard_D4s_v3",
"Standard_D8s_v3"
],
"metadata": {
  "description": "The size of the Bootstrap Virtual Machine"
}
}

"variables": {
  "location": "[resourceGroup().location]",
  "virtualNetworkName": "[concat(parameters('baseName'), '-vnet')]",
  "virtualNetworkID": "[resourceId('Microsoft.Network/virtualNetworks','variables('virtualNetworkName'))]",
  "masterSubnetName": "[concat(parameters('baseName'), '-master-subnet')]",
  "masterSubnetRef": "[concat(variables('virtualNetworkID'), '/subnets/',variables('masterSubnetName'))]",
  "masterLoadBalancerName": "[concat(parameters('baseName'), '-public-lb')]",
  "internalLoadBalancerName": "[concat(parameters('baseName'), '-internal-lb')]",
  "sshKeyPath": "/home/core/.ssh/authorized_keys",
  "identityName": "[concat(parameters('baseName'), '-identity')]",
  "vmName": "[concat(parameters('baseName'), '-bootstrap')]",
  "nicName": "[concat(variables('vmName'), '-nic')]",
  "imageName": "[concat(parameters('baseName'), '-image')]",
  "clusterNsgName": "[concat(parameters('baseName'), '-nsg')]",
  "sshPublicIpAddressName": "[concat(variables('vmName'), '-ssh-pip')]"
},
"resources": [
  
  "apiVersion": "2018-12-01",
  "type": "Microsoft.Network/publicIPAddresses",
  "name": "[variables('sshPublicIpAddressName')]",
  "location": "[variables('location')]",
  "sku": {
    "name": "Standard"
  },
  "properties": {
    "publicIPAddressMethod": "Static",
    "dnsSettings": {
      "domainNameLabel": "[variables('sshPublicIpAddressName')]"
    }
  }
],

"apiVersion": "2018-06-01",

CHAPTER 3. INSTALLING ON AZURE
"type" : "Microsoft.Network/networkInterfaces",
"name" : "[variables('nicName')]",
"location" : "[variables('location')]",
"dependsOn" : [
  "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]
],
"properties" : {
  "ipConfigurations" : [
    {
      "name" : "pipConfig",
      "properties" : {
        "privateIPAddressAllocationMethod" : "Dynamic",
        "publicIPAddress" : {
          "id" : "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
        },
        "subnet" : {
          "id" : "[variables('masterSubnetRef')]"
        },
        "loadBalancerBackendAddressPools" : [
          {
            "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
          },
          {
            "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
          }
        ]
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Compute/virtualMachines",
      "name" : "[variables('vmName')]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceId('Microsoft.ManagedIdentity/userAssignedIdentities', variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('bootstrapVMSize')]"
        },
        "osProfile" : {
"computerName" : "[variables('vmName')]",
"adminUsername" : "core",
"customData" : "[parameters('bootstrapIgnition')]",
"linuxConfiguration" : {
  "disablePasswordAuthentication" : true,
  "ssh" : {
    "publicKeys" : [
      {
        "path" : "[variables('sshKeyPath')]",
        "keyData" : "[parameters('sshKeyData')]"
      }
    ]
  }
},
"storageProfile" : {
  "imageReference" : {
    "id" : "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
  },
  "osDisk" : {
    "name" : "[concat(variables('vmName'), '_OSDisk')]",
    "osType" : "Linux",
    "createOption" : "FromImage",
    "managedDisk" : {
      "storageAccountType" : "Premium_LRS"
    },
    "diskSizeGB" : 100
  }
},
"networkProfile" : {
  "networkInterfaces" : [
    {
      "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
    }
  ]
},
{
  "apiVersion" : "2018-06-01",
  "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
  "location" : "[variables('location')]",
  "dependsOn" : [
    "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
  ],
  "properties" : {
    "protocol" : "Tcp",
    "sourcePortRange" : "***",
    "destinationPortRange" : "22",
    "sourceAddressPrefix" : "***",
    "destinationAddressPrefix" : "***",
    "access" : "Allow",
    "priority" : 100,
    "direction" : "Inbound"
  }
}
3.9.15. Creating the control plane machines in Azure

You must create the control plane machines in Microsoft Azure for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.

**NOTE**
If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.
- Create the bootstrap machine.

**Procedure**

1. Copy the template from the ARM template for control plane machines section of this topic and save it as `05_masters.json` in your cluster’s installation directory. This template describes the control plane machines that your cluster requires.

2. Export the following variable needed by the control plane machine deployment:

   ```bash
   $ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '
'
   ``

3. Create the deployment by using the `az` CLI:

   ```bash
   $ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/05_masters.json" \
   --parameters masterIgnition="${MASTER_IGNITION}" \ 1
   --parameters sshKeyData="${SSH_KEY}" \ 2
   --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 3
   --parameters baseName="${INFRA_ID}" \ 4
   ``

   **1** The Ignition content for the control plane nodes (also known as the master nodes).
   **2** The SSH RSA public key file as a string.
The name of the private DNS zone to which the control plane nodes are attached.

The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3.9.15.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 3.5. 05_masters.json ARM template

```json
{
   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
   "contentVersion": "1.0.0.0",
   "parameters": {
      "baseName": {
         "type": "string",
         "minLength": 1,
         "metadata": {
            "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
         }
      },
      "masterIgnition": {
         "type": "string",
         "metadata": {
            "description": "Ignition content for the master nodes"
         }
      },
      "numberOfMasters": {
         "type": "int",
         "defaultValue": 3,
         "minValue": 2,
         "maxValue": 30,
         "metadata": {
            "description": "Number of OpenShift masters to deploy"
         }
      },
      "sshKeyData": {
         "type": "securestring",
         "metadata": {
            "description": "SSH RSA public key file as a string"
         }
      },
      "privateDNSZoneName": {
         "type": "string",
         "metadata": {
            "description": "Name of the private DNS zone the master nodes are going to be attached to"
         }
      },
      "masterVMSize": {
         "type": "string",
         "defaultValue": "Standard_D8s_v3",
         "allowedValues": [
            "Standard_A2",
```
"Standard_A3",
"Standard_A4",
"Standard_A5",
"Standard_A6",
"Standard_A7",
"Standard_A8",
"Standard_A9",
"Standard_A10",
"Standard_A11",
"Standard_D2",
"Standard_D3",
"Standard_D4",
"Standard_D11",
"Standard_D12",
"Standard_D13",
"Standard_D14",
"Standard_D2_v2",
"Standard_D3_v2",
"Standard_D4_v2",
"Standard_D5_v2",
"Standard_D8_v3",
"Standard_D11_v2",
"Standard_D12_v2",
"Standard_D13_v2",
"Standard_D14_v2",
"Standard_E2_v3",
"Standard_E4_v3",
"Standard_E8_v3",
"Standard_E16_v3",
"Standard_E32_v3",
"Standard_E64_v3",
"Standard_E2s_v3",
"Standard_E4s_v3",
"Standard_E8s_v3",
"Standard_E16s_v3",
"Standard_E32s_v3",
"Standard_E64s_v3",
"Standard_G1",
"Standard_G2",
"Standard_G3",
"Standard_G4",
"Standard_G5",
"Standard_DS2",
"Standard_DS3",
"Standard_DS4",
"Standard_DS11",
"Standard_DS12",
"Standard_DS13",
"Standard_DS14",
"Standard_DS2_v2",
"Standard_DS3_v2",
"Standard_DS4_v2",
"Standard_DS5_v2",
"Standard_DS11_v2",
"Standard_DS12_v2",
"Standard_DS13_v2"
"Standard_DS14_v2",
"Standard_GS1",
"Standard_GS2",
"Standard_GS3",
"Standard_GS4",
"Standard_GS5",
"Standard_D2s_v3",
"Standard_D4s_v3",
"Standard_D8s_v3"
],
"metadata" : {
   "description" : "The size of the Master Virtual Machines"
}
},
"diskSizeGB" : {
   "type" : "int",
   "defaultValue" : 1024,
   "metadata" : {
      "description" : "Size of the Master VM OS disk, in GB"
   }
}
}
}
},
"variables" : {
   "location" : "[resourceGroup().location]",
   "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
   "virtualNetworkID" : "resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))",
   "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
   "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
   "masterLoadBalancerName" : "[concat(parameters('baseName'), '-public-lb')]",
   "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
   "sshKeyPath" : "/home/core/.ssh/authorized_keys",
   "identityName" : "[concat(parameters('baseName'), '-identity')]",
   "imageName" : "[concat(parameters('baseName'), '-image')]",
   "copy" : [
      {
         "name" : "vmNames",
         "count" : "[parameters('numberOfMasters')]",
         "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
      }
   ]
},
"resources" : [
{
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Network/networkInterfaces",
   "copy" : {
      "name" : "nicCopy",
      "count" : "[length(variables('vmNames'))]
   },
   "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
   "location" : "[variables('location')]",
   "properties" : {
      "ipConfigurations" : [
         {
            .....
         }
      ]
   }
}
"name" : "pipConfig",
"properties" : {
  "privateIPAllocationMethod" : "Dynamic",
  "subnet" : {
    "id" : "[variables('masterSubnetRef')]"
  },
  "loadBalancerBackendAddressPools" : [
    {
      "id" : "[concat('subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('masterLoadBalancerName'), '/backendAddressPools/public-lb-backend')]"
    },
    {
      "id" : "[concat('subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
    }
  ],
  "apiVersion" : "2018-09-01",
  "type" : "Microsoft.Network/privateDnsZones/SRV",
  "name" : "[concat(parameters('privateDNSZoneName'), '/_etcd-server-ssl._tcp')]",
  "location" : "[variables('location')]",
  "properties" : {
    "ttl" : 60,
    "copy" : [
      {
        "name" : "srvRecords",
        "count" : "[length(variables('vmNames'))]",
        "input" : {
          "priority" : 0,
          "weight" : 10,
          "port" : 2380,
          "target" : "[concat('etcd-', copyIndex('srvRecords'), '.', parameters('privateDNSZoneName'))]"
        }
      }
    ],
    "apiVersion" : "2018-09-01",
    "type" : "Microsoft.Network/privateDnsZones/A",
    "copy" : {
      "name" : "dnsCopy",
      "count" : "[length(variables('vmNames'))]"
    },
    "name" : "[concat(parameters('privateDNSZoneName'), '/etcd-', copyIndex())]",
    "location" : "[variables('location')]",
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', variables('vmNames')[copyIndex()], 'nic')]"
    ]
  }
}
"properties": {
  "ttl": 60,
  "aRecords": [
    {
      "ipv4Address": 
      "[reference(concat(variables('vmNames')[copyIndex()], '-', 'nic')).ipConfigurations[0].properties.privateIPAddress"
    }
  ],
  "apiVersion": "2018-06-01",
  "type": "Microsoft.Compute/virtualMachines",
  "copy": {
    "name": "vmCopy",
    "count": "[length(variables('vmNames'))]"
  },
  "name": "[variables('vmNames')[copyIndex()]]",
  "location": "[variables('location')]",
  "identity": {
    "type": "userAssigned",
    "userAssignedIdentities": {
      "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/', variables('identityName'))]": {}
    }
  },
  "dependsOn": [
    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-', 'nic'))],
    "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/A/etcd-', copyIndex())],
    "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/SRV/_etcd-server-ssl._tcp')]]"
  ],
  "properties": {
    "hardwareProfile": {
      "vmSize": "[parameters('masterVMSize')]"
    },
    "osProfile": {
      "computerName": "[variables('vmNames')[copyIndex()]]",
      "adminUsername": "core",
      "customData": "[parameters('masterIgnition')]",
      "linuxConfiguration": {
        "disablePasswordAuthentication": true,
        "ssh": {
          "publicKeys": [
            {
              "path": "[variables('sshKeyPath')]",
              "keyData": "[parameters('sshKeyData')]"
            }
          ]
        }
      }
    },
    "storageProfile": {
      "imageReference": {
        "properties": {
          "ttl": 60,
          "aRecords": [
            {
              "ipv4Address": 
              "[reference(concat(variables('vmNames')[copyIndex()], '-', 'nic')).ipConfigurations[0].properties.privateIPAddress"
            }
          ],
          "apiVersion": "2018-06-01",
          "type": "Microsoft.Compute/virtualMachines",
          "copy": {
            "name": "vmCopy",
            "count": "[length(variables('vmNames'))]"
          },
          "name": "[variables('vmNames')[copyIndex()]]",
          "location": "[variables('location')]",
          "identity": {
            "type": "userAssigned",
            "userAssignedIdentities": {
              "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/', variables('identityName'))]": {}
            }
          },
          "dependsOn": [
            "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-', 'nic'))],
            "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/A/etcd-', copyIndex())],
            "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/SRV/_etcd-server-ssl._tcp')]]"
          ],
          "properties": {
            "hardwareProfile": {
              "vmSize": "[parameters('masterVMSize')]"
            },
            "osProfile": {
              "computerName": "[variables('vmNames')[copyIndex()]]",
              "adminUsername": "core",
              "customData": "[parameters('masterIgnition')]",
              "linuxConfiguration": {
                "disablePasswordAuthentication": true,
                "ssh": {
                  "publicKeys": [
                    {
                      "path": "[variables('sshKeyPath')]",
                      "keyData": "[parameters('sshKeyData')]"
                    }
                  ]
                }
              }
            },
            "storageProfile": {
              "imageReference": {
                "properties": {
                  "ttl": 60,
                  "aRecords": [
                    {
                      "ipv4Address": 
                      "[reference(concat(variables('vmNames')[copyIndex()], '-', 'nic')).ipConfigurations[0].properties.privateIPAddress"
                    }
                  ],
                  "apiVersion": "2018-06-01",
                  "type": "Microsoft.Compute/virtualMachines",
                  "copy": {
                    "name": "vmCopy",
                    "count": "[length(variables('vmNames'))]"
                  },
                  "name": "[variables('vmNames')[copyIndex()]]",
                  "location": "[variables('location')]",
                  "identity": {
                    "type": "userAssigned",
                    "userAssignedIdentities": {
                      "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/', variables('identityName'))]": {}
                    }
                  },
                  "dependsOn": [
                    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-', 'nic'))],
                    "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/A/etcd-', copyIndex())],
                    "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/SRV/_etcd-server-ssl._tcp')]]"
                  ],
                  "properties": {
                    "hardwareProfile": {
                      "vmSize": "[parameters('masterVMSize')]"
                    },
                    "osProfile": {
                      "computerName": "[variables('vmNames')[copyIndex()]]",
                      "adminUsername": "core",
                      "customData": "[parameters('masterIgnition')]",
                      "linuxConfiguration": {
                        "disablePasswordAuthentication": true,
                        "ssh": {
                          "publicKeys": [
                            {
                              "path": "[variables('sshKeyPath')]",
                              "keyData": "[parameters('sshKeyData')]"
                            }
                          ]
                        }
                      }
                    },
                    "storageProfile": {
                      "imageReference": {
                        "properties": {
                          "ttl": 60,
                          "aRecords": [
                            {
                              "ipv4Address": 
                              "[reference(concat(variables('vmNames')[copyIndex()], '-', 'nic')).ipConfigurations[0].properties.privateIPAddress"
                            }
                          ],
                          "apiVersion": "2018-06-01",
                          "type": "Microsoft.Compute/virtualMachines",
                          "copy": {
                            "name": "vmCopy",
                            "count": "[length(variables('vmNames'))]"
                          },
                          "name": "[variables('vmNames')[copyIndex()]]",
                          "location": "[variables('location')]",
                          "identity": {
                            "type": "userAssigned",
                            "userAssignedIdentities": {
                              "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/', variables('identityName'))]": {}
                            }
                          },
                          "dependsOn": [
                            "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-', 'nic'))],
                            "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/A/etcd-', copyIndex())],
                            "[concat('Microsoft.Network/privateDnsZones/', parameters('privateDNSZoneName'), '/SRV/_etcd-server-ssl._tcp')]]"
                          ],
                          "properties": {
                            "hardwareProfile": {
                              "vmSize": "[parameters('masterVMSize')]"
                            },
                            "osProfile": {
                              "computerName": "[variables('vmNames')[copyIndex()]]",
                              "adminUsername": "core",
                              "customData": "[parameters('masterIgnition')]",
                              "linuxConfiguration": {
                                "disablePasswordAuthentication": true,
                                "ssh": {
                                  "publicKeys": [
                                    {
                                      "path": "[variables('sshKeyPath')]",
                                      "keyData": "[parameters('sshKeyData')]"
                                    }
                                  ]
                                }
                              }
                          }
                        }
                      }
                    }
                  }
                }
            }
          }
        }
      }
    }
  }
}
3.9.16. Wait for bootstrap completion and remove bootstrap resources in Azure

After you create all of the required infrastructure in Microsoft Azure, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
- Create and configure networking and load balancers in Azure.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

**Procedure**
1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \
   --log-level info
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name ${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${INFRA_ID}-bootstrap-ssh-pip
```

### 3.9.17. Creating additional worker machines in Azure

You can create worker machines in Microsoft Azure for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type `06_workers.json` in the file.

**NOTE**

If you do not use the provided ARM template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure an Azure account.
- Generate the Ignition config files for your cluster.
- Create and configure a VNet and associated subnets in Azure.
Create and configure networking and load balancers in Azure.

Create control plane and compute roles.

Create the bootstrap machine.

Create the control plane machines.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster’s installation directory. This template describes the worker machines that your cluster requires.

2. Export the following variable needed by the worker machine deployment:

   ```bash
   $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d \"n\`
   ```

3. Create the deployment by using the **az CLI**:

   ```bash
   $ az deployment group create -g ${RESOURCE_GROUP} \
   --template-file "<installation_directory>/06_workers.json" \
   --parameters workerIgnition="${WORKER_IGNITION}" \  
   --parameters sshKeyData="${SSH_KEY}" \  
   --parameters baseName="${INFRA_ID}"
   ```

   **1**  The Ignition content for the worker nodes.

   **2**  The SSH RSA public key file as a string.

   **3**  The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3.9.17.1. **ARM template for worker machines**

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

**Example 3.6. 06_workers.json ARM template**

```json
{
   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
   "contentVersion": "1.0.0.0",
   "parameters": {
      "baseName": {
         "type": "string",
         "minLength": 1,
         "metadata": {
            "description": "Base name to be used in resource names (usually the cluster’s Infra ID)"
         }
      },
      "workerIgnition": {
         "type": "string",
         "metadata": {
         }
      }
   }
}
```
"description": "Ignition content for the worker nodes"
},

"numberOfNodes": {
  "type": "int",
  "defaultValue": 3,
  "minValue": 2,
  "maxValue": 30,
  "metadata": {
    "description": "Number of OpenShift compute nodes to deploy"
  }
},

"sshKeyData": {
  "type": "securestring",
  "metadata": {
    "description": "SSH RSA public key file as a string"
  }
},

"nodeVMSize": {
  "type": "string",
  "defaultValue": "Standard_D4s_v3",
  "allowedValues": [
    "Standard_A2",
    "Standard_A3",
    "Standard_A4",
    "Standard_A5",
    "Standard_A6",
    "Standard_A7",
    "Standard_A8",
    "Standard_A9",
    "Standard_A10",
    "Standard_A11",
    "Standard_D2",
    "Standard_D3",
    "Standard_D4",
    "Standard_D11",
    "Standard_D12",
    "Standard_D13",
    "Standard_D14",
    "Standard_D2_v2",
    "Standard_D3_v2",
    "Standard_D4_v2",
    "Standard_D5_v2",
    "Standard_D8_v3",
    "Standard_D11_v2",
    "Standard_D12_v2",
    "Standard_D13_v2",
    "Standard_D14_v2",
    "Standard_E2_v3",
    "Standard_E4_v3",
    "Standard_E8_v3",
    "Standard_E16_v3",
    "Standard_E32_v3",
    "Standard_E64_v3",
    "Standard_E2s_v3",
    "Standard_E4s_v3"
"Standard_E8s_v3",
"Standard_E16s_v3",
"Standard_E32s_v3",
"Standard_E64s_v3",
"Standard_G1",
"Standard_G2",
"Standard_G3",
"Standard_G4",
"Standard_G5",
"Standard_DS2",
"Standard_DS3",
"Standard_DS4",
"Standard_DS11",
"Standard_DS12",
"Standard_DS13",
"Standard_DS14",
"Standard_DS2_v2",
"Standard_DS3_v2",
"Standard_DS4_v2",
"Standard_DS5_v2",
"Standard_DS11_v2",
"Standard_DS12_v2",
"Standard_DS13_v2",
"Standard_DS14_v2",
"Standard_GS1",
"Standard_GS2",
"Standard_GS3",
"Standard_GS4",
"Standard_GS5",
"Standard_D2s_v3",
"Standard_D4s_v3",
"Standard_D8s_v3"
],
"metadata": {
  "description": "The size of the each Node Virtual Machine"
}

"variables": {
  "location": ":[resourceGroup().location],
  "virtualNetworkName": [concat(parameters('baseName'), '-vnet')],
  "virtualNetworkID": [resourceId('Microsoft.Network/virtualNetworks', variables('virtualNetworkName'))],
  "nodeSubnetName": [concat(parameters('baseName'), '-worker-subnet')],
  "nodeSubnetRef": [concat(variables('virtualNetworkID'), '/subnets/', variables('nodeSubnetName'))],
  "infraLoadBalancerName": [parameters('baseName')],
  "sshKeyPath": "/home/capi/.ssh/authorized_keys",
  "identityName": [concat(parameters('baseName'), '-identity')],
  "imageName": [concat(parameters('baseName'), '-image')],
  "copy": [{
    "name": "vmNames",
    "count": [parameters('numberOfNodes')],
    "input": [concat(parameters('baseName'), '-worker-', variables('location'), '-', copyIndex('vmNames', 1))]
  }]
"resources" : [
    {
        "apiVersion" : "2019-05-01",
        "name" : "[concat('node', copyIndex())]",
        "type" : "Microsoft.Resources/deployments",
        "copy" : {
            "name" : "nodeCopy",
            "count" : "[length(variables('vmNames'))]"
        },
        "properties" : {
            "mode" : "Incremental",
            "template" : {
                "contentVersion" : "1.0.0.0",
                "resources" : [
                    {
                        "apiVersion" : "2018-06-01",
                        "type" : "Microsoft.Network/networkInterfaces",
                        "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
                        "location" : "[variables('location')]",
                        "properties" : {
                            "ipConfigurations" : [
                                {
                                    "name" : "pipConfig",
                                    "properties" : {
                                        "privateIPAddressAllocationMethod" : "Dynamic",
                                        "subnet" : {
                                            "id" : "[variables('nodeSubnetRef')]"
                                        }
                                    }
                                }
                            ]
                        }
                    },
                    {
                        "apiVersion" : "2018-06-01",
                        "type" : "Microsoft.Compute/virtualMachines",
                        "name" : "[variables('vmNames')[copyIndex()]]",
                        "location" : "[variables('location')]",
                        "tags" : {
                            "kubernetes.io-cluster-ffranzupi" : "owned"
                        },
                        "identity" : {
                            "type" : "userAssigned",
                            "userAssignedIdentities" : {
                                "[resourceId('Microsoft.ManagedIdentity/userAssignedIdentities/', variables('identityName'))]" : {}
                            }
                        },
                        "dependsOn" : [
                            "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames') [copyIndex()], '-nic'))]"
                        ]
                    }
                ]
            }
        }
    }
]
"properties": {
  "hardwareProfile": {
    "vmSize": "[parameters('nodeVMSize')]"
  },
  "osProfile": {
    "computerName": "[variables('vmNames')[copyIndex()]]",
    "adminUsername": "capi",
    "customData": "[parameters('workerIgnition')]",
    "linuxConfiguration": {
      "disablePasswordAuthentication": true,
      "ssh": {
        "publicKeys": [
          {
            "path": "[variables('sshKeyPath')]",
            "keyData": "[parameters('sshKeyData')]"
          }
        ]
      }
    }
  },
  "storageProfile": {
    "imageReference": {
      "id": "[resourceId('Microsoft.Compute/images', variables('imageName'))]"
    },
    "osDisk": {
      "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
      "osType": "Linux",
      "createOption": "FromImage",
      "managedDisk": {
        "storageAccountType": "Premium_LRS"
      },
      "diskSizeGB": 128
    }
  },
  "networkProfile": {
    "networkInterfaces": [
      {
        "id": "[resourceId('Microsoft.Network/networkInterfaces',
        concat(variables('vmNames')[copyIndex()], '-nic'))]",
        "properties": {
          "primary": true
        }
      }
    ]
  }
}
3.9.18. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

3.9.18.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xzvf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

3.9.18.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   $ oc <command>
After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 3.9.18.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

#### Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your PATH.
   
   To check your PATH, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 3.9.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

#### Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

#### Procedure

1. Export the `kubeadmin` credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

   - For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

**Example output**

```
system:admin
```

### 3.9.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

**Example output**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

```
NAME        AGE     REQUESTOR                                                                 CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...```

---

**CHAPTER 3. INSTALLING ON AZURE**

---

679
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

   **NOTE**

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

   **NOTE**

   For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

   - To approve them individually, run the following command for each valid CSR:

     ```bash
     $ oc adm certificate approve <csr_name>  # 1
     ```

     **1**  
     `<csr_name>` is the name of a CSR from the list of current CSRs.

   - To approve all pending CSRs, run the following command:

     ```bash
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
     ```

     **NOTE**

     Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

   ```bash
   $ oc get csr
   ```

   **Example output**
5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name>
  ```

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{\n}}{{end}}{{end}}' | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the Ready status. Verify this by running the following command:

  ```bash
  $ oc get nodes
  ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

Additional information

- For more information on CSRs, see Certificate Signing Requests.

3.9.21. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard `*.apps.(baseDomain).` or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites
You deployed an OpenShift Container Platform cluster on Microsoft Azure by using infrastructure that you provisioned.

- Install the OpenShift CLI (`oc`).
- Install the `jq` package.
- Install or update the Azure CLI.

**Procedure**

1. Confirm the Ingress router has created a load balancer and populated the `EXTERNAL-IP` field:
   ```bash
   $ oc -n openshift-ingress get service router-default
   
   Example output
   
   NAME          TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
   router-default LoadBalancer  172.30.20.10   35.130.120.110 80:32288/TCP,443:31215/TCP   20s
   ```

2. Export the Ingress router IP as a variable:
   ```bash
   $ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
   
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
   $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
   $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
   ```

3. Add a `*.apps` record to the public DNS zone.
   a. If you are adding this cluster to a new public zone, run:
      ```bash
      $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
      ```
   b. If you are adding this cluster to an already existing public zone, run:
      ```bash
      $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
      ```

4. Add a `*.apps` record to the private DNS zone:
   a. Create a `*.apps` record by using the following command:
      ```bash
      $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
      ```
   b. Add the `*.apps` record to the private DNS zone by using the following command:
      ```bash
      $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
      ```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster’s current routes:
3.9.22. Completing an Azure installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

**Prerequisites**

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure infrastructure.
- Install the `oc` CLI and log in.

**Procedure**

- Complete the cluster installation:

  ```bash
  $ ./openshift-install --dir <installation_directory> wait-for install-complete
  ```

  **Example output**

  ```bash
  INFO Waiting up to 30m0s for the cluster to initialize...
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.9.23. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

3.10. UNINSTALLING A CLUSTER ON AZURE

You can remove a cluster that you deployed to Microsoft Azure.

3.10.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

- Have a copy of the installation program that you used to deploy the cluster.
- Have the files that the installation program generated when you created your cluster.

Procedure
1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. To view different details, specify `warn`, `debug`, or `error` instead of `info`.

NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.
CHAPTER 4. INSTALLING ON GCP

4.1. CONFIGURING A GCP PROJECT

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

4.1.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See Creating and Managing Projects in the GCP documentation.

**IMPORTANT**

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the `api-int.<cluster_name>.<base_domain>` URL; the Premium Tier is required for internal load balancing.

4.1.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. See Enabling services in the GCP documentation.

**Table 4.1. Required API services**

<table>
<thead>
<tr>
<th>API service</th>
<th>Console service name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute Engine API</td>
<td>compute.googleapis.com</td>
</tr>
<tr>
<td>Google Cloud APIs</td>
<td>cloudapis.googleapis.com</td>
</tr>
<tr>
<td>Cloud Resource Manager API</td>
<td>cloudresourcemanager.googleapis.com</td>
</tr>
<tr>
<td>Google DNS API</td>
<td>dns.googleapis.com</td>
</tr>
</tbody>
</table>
4.1.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.

   **NOTE**

   If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](https://domains.google).  

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](https://cloud.google.com/dns/docs/public-zones) in the GCP documentation.  

   Use an appropriate root domain, such as `openshiftcorp.com`, or subdomain, such as `clusters.openshiftcorp.com`.

3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](https://cloud.google.com/dns/docs/authoritative-name-servers) in the GCP documentation.  

   You typically have four name servers.

4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](https://domains.google/help/01051049).  

5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](https://cloud.google.com/dns/docs/migrating) in the GCP documentation.
6. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain. This process might include a request to your company’s IT department or the division that controls the root domain and DNS services for your company.

### 4.1.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default Quotas do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

**Table 4.2. GCP resources used in a default cluster**

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Location</th>
<th>Total resources required</th>
<th>Resources removed after bootstrap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service account</td>
<td>IAM</td>
<td>Global</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Firewall rules</td>
<td>Compute</td>
<td>Global</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>Forwarding rules</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>In-use global IP addresses</td>
<td>Compute</td>
<td>Global</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Health checks</td>
<td>Compute</td>
<td>Global</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Images</td>
<td>Compute</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Networks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Static IP addresses</td>
<td>Compute</td>
<td>Region</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Routers</td>
<td>Compute</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routes</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Subnetworks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Target pools</td>
<td>Compute</td>
<td>Global</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>CPUs</td>
<td>Compute</td>
<td>Region</td>
<td>28</td>
<td>4</td>
</tr>
<tr>
<td>Persistent disk SSD (GB)</td>
<td>Compute</td>
<td>Region</td>
<td>896</td>
<td>128</td>
</tr>
</tbody>
</table>
NOTE

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- asia-east2
- asia-northeast2
- asia-south1
- australia-southeast1
- europe-north1
- europe-west2
- europe-west3
- europe-west6
- northamerica-northeast1
- southamerica-east1
- us-west2

You can increase resource quotas from the GCP console, but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

4.1.5. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See Creating a service account in the GCP documentation.

2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the Owner role to it. See Granting roles to a service account for specific resources.
NOTE
While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. Create the service account key in JSON format. See Creating service account keys in the GCP documentation. The service account key is required to create a cluster.

4.1.5.1. Required GCP permissions

When you attach the Owner role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. To deploy an OpenShift Container Platform cluster, the service account requires the following permissions. If you deploy your cluster into an existing VPC, the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Optional roles

For the cluster to create new limited credentials for its Operators, add the following role:

- Service Account Key Admin

The roles are applied to the service accounts that the control plane and compute machines use:

Table 4.3. GCP service account permissions

<table>
<thead>
<tr>
<th>Account</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Plane</td>
<td>roles/compute.instanceAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.networkAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.securityAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
</tbody>
</table>
### 4.1.6. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **australia-southeast1** (Sydney, Australia)
- **europe-north1** (Hamina, Finland)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, Northern Virginia, USA)
- **us-west1** (The Dalles, Oregon, USA)
4.1.7. Next steps

- Install an OpenShift Container Platform cluster on GCP. You can install a customized cluster or quickly install a cluster with default options.

4.2. MANUALLY CREATING IAM FOR GCP

In environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster `kube-system` namespace, you can put the Cloud Credential Operator (CCO) into manual mode before you install the cluster.

4.2.1. Alternatives to storing administrator-level secrets in the `kube-system` project

The Cloud Credential Operator (CCO) manages cloud provider credentials as Kubernetes custom resource definitions (CRDs). You can configure the CCO to suit the security requirements of your organization by setting different values for the `credentialsMode` parameter in the `install-config.yaml` file.

If you prefer not to store an administrator-level credential secret in the cluster `kube-system` project, you can set the `credentialsMode` parameter for the CCO to `Manual` when installing OpenShift Container Platform and manage your cloud credentials manually.

Using manual mode allows each cluster component to have only the permissions it requires, without storing an administrator-level credential in the cluster. You can also use this mode if your environment does not have connectivity to the cloud provider public IAM endpoint. However, you must manually reconcile permissions with new release images for every upgrade. You must also manually supply credentials for every component that requests them.

Additional resources

- Rotating or removing cloud provider credentials.

For a detailed description of all available CCO credential modes and their supported platforms, see the Cloud Credential Operator reference.

4.2.2. Manually create IAM

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster `kube-system` namespace.

Procedure

1. To generate the manifests, run the following command from the directory that contains the installation program:

```bash
...
1 $ openshift-install create manifests --dir <installation_directory>

For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

2. Insert a config map into the manifests directory so that the Cloud Credential Operator is placed in manual mode:

   $ cat <<EOF > mycluster/manifests/cco-configmap.yaml
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: cloud-credential-operator-config
     namespace: openshift-cloud-credential-operator
     annotations:
       release.openshift.io/create-only: "true"
   data:
     disabled: "true"
   EOF

3. Remove the `admin` credential secret created using your local cloud credentials. This removal prevents your `admin` credential from being stored in the cluster:

   $ rm mycluster/openshift/99_cloud-creds-secret.yaml

4. From the directory that contains the installation program, obtain details of the OpenShift Container Platform release image that your `openshift-install` binary is built to use:

   $ openshift-install version

   Example output

   release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64

5. Locate all `CredentialsRequest` objects in this release image that target the cloud you are deploying on:

   $ oc adm release extract quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64 --credentials-requests --cloud=gcp

   This displays the details for each request.

   **Sample `CredentialsRequest` object**

   ```yaml
   apiVersion: cloudcredential.openshift.io/v1
   kind: CredentialsRequest
   metadata:
     labels:
       controller-tools.k8s.io: "1.0"
   name: openshift-image-registry-gcs
   namespace: openshift-cloud-credential-operator
   spec:
   ```
6. Create YAML files for secrets in the `openshift-install` manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the `spec.secretRef` for each `credentialsRequest`. The format for the secret data varies for each cloud provider.

7. From the directory that contains the installation program, proceed with your cluster creation:

   ```bash
   $ openshift-install create cluster --dir <installation_directory>
   ```

4.2.3. Admin credentials root secret format

Each cloud provider uses a credentials root secret in the `kube-system` namespace by convention, which is then used to satisfy all credentials requests and create their respective secrets. This is done either by minting new credentials, with `mint mode`, or by copying the credentials root secret, with `passthrough mode`.

The format for the secret varies by cloud, and is also used for each `CredentialsRequest` secret.

**Google Cloud Platform (GCP) secret format**

```yaml
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
  name: gcp-credentials
stringData:
  service_account.json: <ServiceAccount>
```

4.2.4. Upgrading clusters with manually maintained credentials

If credentials are added in a future release, the Cloud Credential Operator (CCO) **upgradable** status for a cluster with manually maintained credentials changes to **false**. For minor release, for example, from 4.5 to 4.6, this status prevents you from upgrading until you have addressed any updated permissions. For z-stream releases, for example, from 4.5.10 to 4.5.11, the upgrade is not blocked, but the credentials must still be updated for the new release.
Use the Administrator perspective of the web console to determine if the CCO is upgradeable.

1. Navigate to Administration → Cluster Settings.

2. To view the CCO status details, click cloud-credential in the Cluster Operators list.

3. If the Upgradeable status in the Conditions section is False, examine the credentialsRequests for the new release and update the manually maintained credentials on your cluster to match before upgrading.

In addition to creating new credentials for the release image that you are upgrading to, you must review the required permissions for existing credentials and accommodate any new permissions requirements for existing components in the new release. The CCO cannot detect these mismatches and will not set upgradable to false in this case.

The Manually creating IAM section of the installation content for your cloud provider explains how to obtain and use the credentials required for your cloud.

4.2.5. Mint mode

Mint mode is the default and recommended Cloud Credential Operator (CCO) credentials mode for OpenShift Container Platform. In this mode, the CCO uses the provided administrator-level cloud credential to run the cluster. Mint mode is supported for AWS, GCP, and Azure.

In mint mode, the admin credential is stored in the kube-system namespace and then used by the CCO to process the CredentialsRequest objects in the cluster and create users for each with specific permissions.

The benefits of mint mode include:

- Each cluster component has only the permissions it requires
- Automatic, on-going reconciliation for cloud credentials, including additional credentials or permissions that might be required for upgrades

One drawback is that mint mode requires admin credential storage in a cluster kube-system secret.

4.2.6. Mint Mode with removal or rotation of the admin credential

Currently, this mode is only supported on AWS.

In this mode, a user installs OpenShift Container Platform with an admin credential just like the normal mint mode. However, this mode removes the admin credential secret from the cluster post-installation.

The administrator can have the Cloud Credential Operator make its own request for a read-only credential that allows it to verify if all CredentialsRequest objects have their required permissions, thus the admin credential is not required unless something needs to be changed. After the associated credential is removed, it can be destroyed on the underlying cloud, if desired.

Prior to upgrade, the admin credential should be restored. In the future, upgrade might be blocked if the credential is not present.

The admin credential is not stored in the cluster permanently.

This mode still requires the admin credential in the cluster for brief periods of time. It also requires manually re-instating the secret with admin credentials for each upgrade.
4.2.7. Next steps

- Install an OpenShift Container Platform cluster:
  - Installing a cluster quickly on GCP with default options on installer-provisioned infrastructure
  - Install a cluster with cloud customizations on installer-provisioned infrastructure
  - Install a cluster with network customizations on installer-provisioned infrastructure

4.3. INSTALLING A CLUSTER QUICKLY ON GCP

In OpenShift Container Platform version 4.6, you can install a cluster on Google Cloud Platform (GCP) that uses the default configuration options.

4.3.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

4.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.3.3. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N " " -f <path>/<file_name>
   ```

   1. Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```bash
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.
3. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/file_name
```

**Example output**

```
Identity added: /home/<you>/<path>/file_name (<computer_name>)
```

|   | Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa |

4. Set the **GOOGLE_APPLICATION_CREDENTIALS** environment variable to the full path to your service account private key file.

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
```

5. Verify that the credentials were applied.

```
$ gcloud auth list
```

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 4.3.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 4.3.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:

   - The `GOOGLE_CREDENTIALS`, `GOOGLE_CLOUD_KEYFILE_JSON`, or `GCloud_KEYFILE_JSON` environment variables
   - The `~/.gcp/osServiceAccount.json` file
   - The `gcloud cli` default credentials

2. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> --log-level=info
```

For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
To view different installation details, specify warn, debug, or error instead of info.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Provide values at the prompts:

a. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

b. Select gcp as the platform to target.

c. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

d. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

e. Select the region to deploy the cluster to.

f. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

g. Enter a descriptive name for your cluster. If you provide a name that is longer than 6 characters, only the first 6 characters will be used in the infrastructure ID that is generated from the cluster name.

h. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the kubeadmin user, display in your terminal.

**Example output**

```
...
```
NOTE

The cluster access and credential information also outputs to 
<installation_directory>/openshift_install.log when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.

- If you assigned the Owner role to your service account, you can remove that role and replace it with the Viewer role.

- If you included the Service Account Key Admin role, you can remove it.

4.3.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.
4.3.6.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```bash
   $ tar xvzf <file>
   ```

5. Place the `oc` binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```bash
$ oc <command>
```

4.3.6.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   ```cmd
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```cmd
C:\> oc <command>
```

4.3.6.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.
Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```bash
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```bash
   $ oc <command>
   ```

4.3.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

Additional resources
4.3.8. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

4.3.9. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

4.4. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a customized cluster on infrastructure that the installation program provisions on Google Cloud Platform (GCP). To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

4.4.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

4.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
4.4.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ``

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:
NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the **ssh-agent**:

   ```$ ssh-add <path>/<file_name>```

   **Example output**

   ```Identity added: /home/<you>/<path>/<file_name> (<computer_name>)```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

4. Set the **GOOGLE_APPLICATION_CREDENTIALS** environment variable to the full path to your service account private key file.

   ```$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"```

5. Verify that the credentials were applied.

   ```$ gcloud auth list```

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**4.4.4. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 4.4.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   **1**

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select gcp as the platform to target.

iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

v. Select the region to deploy the cluster to.

vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

vii. Enter a descriptive name for your cluster.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the install-config.yaml file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the install-config.yaml file so that you can use it to install multiple clusters.

IMPORTANT

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

4.4.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.
NOTE
After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT
The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

4.4.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 4.4. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}</code>, <code>{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws**, **baremetal**, **azure**, **openstack**, **ovirt**, **vsphere**. For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td>installation: <strong>aws</strong>, <strong>baremetal</strong>, <strong>azure</strong>, <strong>openstack</strong>, <strong>ovirt</strong>,</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>vsphere</strong>. For additional information about <code>platform.&lt;platform&gt;</code> parameters,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>consult the following table for your specific platform.</td>
<td></td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate</td>
<td>`{</td>
</tr>
<tr>
<td></td>
<td>downloading container images for OpenShift Container Platform components</td>
<td>&quot;auths&quot;:{</td>
</tr>
<tr>
<td></td>
<td>from services such as Quay.io.</td>
<td>&quot;cloud.openshift.com&quot;:{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth&quot;:&quot;b3Blb=&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;quay.io&quot;:{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth&quot;:&quot;b3Blb=&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>

4.4.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 4.5. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td>for the cluster</td>
<td></td>
</tr>
<tr>
<td></td>
<td>network.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
networking.machineNetwork.cidr

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**
Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

4.4.5.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 4.6. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
### credentialsMode

The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fips</code></td>
<td>Sources and repositories for the release-image content.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td><code>imageContentSources</code></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><code>imageContentSources.source</code></td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td><code>imageContentSources.mirrors</code></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td><code>publish</code></td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
### sshKey

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

---

### 4.4.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.gcp.network</code></td>
<td>The name of the existing VPC that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.gcp.region</code></td>
<td>The name of the GCP region that hosts your cluster.</td>
<td>Any valid region name, such as <code>us-central1</code>.</td>
</tr>
<tr>
<td><code>platform.gcp.type</code></td>
<td>The GCP machine type.</td>
<td>The GCP machine type.</td>
</tr>
<tr>
<td><code>platform.gcp.zones</code></td>
<td>The availability zones where the installation program creates machines for the specified MachinePool.</td>
<td>A list of valid GCP availability zones, such as <code>us-central1-a</code>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>platform.gcp.controlPlaneSubnet</code></td>
<td>The name of the existing subnet in your VPC that you want to deploy your control plane machines to.</td>
<td>The subnet name.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.gcp.computeSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your compute machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.licenses</td>
<td>A list of license URLs that must be applied to the compute images.</td>
<td>Any license available with the license API, such as the license to enable nested virtualization. You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installer to copy the source image before use.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.disk SizeGB</td>
<td>The size of the disk in gigabytes (GB).</td>
<td>Any size between 16 GB and 65536 GB.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.disk Type</td>
<td>The type of disk.</td>
<td>Either the default <strong>pd-ssd</strong> or the <strong>pd-standard</strong> disk type. The control plane nodes must be the <strong>pd-ssd</strong> disk type. The worker nodes can be either type.</td>
</tr>
</tbody>
</table>

#### 4.4.5.2. Sample customized install-config.yaml file for GCP

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane:
  hyperthreading: Enabled
  name: master
platform:
gcp:
```

---

---
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, `-`, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous
multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as `n1-standard-8`, for your machines if you disable simultaneous multithreading.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

You can optionally provide the `sshKey` value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 4.4.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  
     noProxy: example.com
   additionalTrustBundle:
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
   4 If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 4.4.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
   - The `GOOGLE_CREDENTIALS`, `GOOGLE_CLOUD_KEYFILE_JSON`, or `GCLOUD_KEYFILE_JSON` environment variables
   - The `~/.gcp/osServiceAccount.json` file
   - The `gcloud cli` default credentials

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info  
   
   1
   ```

   1 For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.
When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.

- If you assigned the Owner role to your service account, you can remove that role and replace it with the Viewer role.
- If you included the Service Account Key Admin role, you can remove it.

### 4.4.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

4.4.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

4.4.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
4.4.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

4.4.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami
4.4.9. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

4.4.10. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

4.5. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Google Cloud Platform (GCP). By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

You must set most of the network configuration parameters during installation, and you can modify only kubeProxy configuration parameters in a running cluster.

4.5.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.
4.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.5.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.
Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
   
   **NOTE**
   
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add /path/to/file
   
   Example output
   
   Identity added: /home/<you>/path/to/file (<computer_name>)
   
   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

4. Set the `GOOGLE_APPLICATION_CREDENTIALS` environment variable to the full path to your service account private key file.

   ```bash
   $ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
   
   5. Verify that the credentials were applied.

   ```bash
   $ gcloud auth list
   
   Next steps
   
   - When you install OpenShift Container Platform, provide the SSH public key to the installation program.

4.5.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.
Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**
   
   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**
   
   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 4.5.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create the `install-config.yaml` file.
a. Change to the directory that contains the installation program and run the following command:

```bash
$ ./openshift-install create install-config --dir <installation_directory>  # 1
```

For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `gcp` as the platform to target.

iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

v. Select the region to deploy the cluster to.

vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

vii. Enter a descriptive name for your cluster.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the `Installation configuration parameters` section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
4.5.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 4.5.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiVersion</code></td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td><code>baseDomain</code></td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;..&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{   &quot;auths&quot;:{   &quot;cloud.openshift.com&quot;:{   &quot;auth&quot;:&quot;b3Blb=&quot;&quot;,   &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;   },   &quot;quay.io&quot;:{   &quot;auth&quot;:&quot;b3Blb=&quot;&quot;,   &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;   }   } }</td>
</tr>
</tbody>
</table>

### 4.5.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 4.9. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>

732
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong> You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
<td></td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either <code>OpenShiftSDN</code> or <code>OVNKubernetes</code>. The default value is <code>OpenShiftSDN</code>.</td>
</tr>
<tr>
<td>Type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>10.128.0.0/14</code> with a host prefix of <code>/23</code>.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>- clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: <code>10.128.0.0/14</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- hostPrefix: <code>23</code></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between <code>0</code> and <code>32</code>.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to <code>23</code> then each node is assigned a <code>/23</code> subnet out of the given cidr. A <code>hostPrefix</code> value of <code>23</code> provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>23</code>.</td>
<td></td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <code>172.30.0.0/16</code>.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>172.30.0.0/16</code></td>
</tr>
</tbody>
</table>
The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

An array of objects. For example:

```
networking:
  machineNetwork:
    - cidr: 10.0.0.0/16
```

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**
Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

### 4.5.5.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 4.10. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>additionalTrustBundle</code></td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td><code>compute</code></td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>compute.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or *hyperthreading*, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hyperthreading</em>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or {}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td>dynamically tries to determine the capabilities of the provided credentials,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td>IMPORTANT</td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.</td>
<td></td>
</tr>
<tr>
<td>NOTE</td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
4.5.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

**Table 4.11. Additional GCP parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><strong>Internal</strong> or <strong>External</strong>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <strong>Internal</strong>. The default value is <strong>External</strong>.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example:</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your <code>ssh-agent</code> process uses.</td>
<td></td>
</tr>
</tbody>
</table>

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

**Parameter**

<table>
<thead>
<tr>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>The name of the existing VPC that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td>The name of the GCP region that hosts your cluster.</td>
<td>Any valid region name, such as <code>us-central1</code>.</td>
</tr>
<tr>
<td>The <strong>GCP machine type</strong>.</td>
<td>The <strong>GCP machine type</strong>.</td>
</tr>
<tr>
<td>The availability zones where the installation program creates machines for the specified MachinePool.</td>
<td>A list of valid <strong>GCP availability zones</strong>, such as <code>us-central1-a</code>, in a <strong>YAML sequence</strong>.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.gcp.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your control plane machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.computeSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your compute machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.licenses</td>
<td>A list of license URLs that must be applied to the compute images.</td>
<td>Any license available with the license API, such as the license to enable nested virtualization. You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installer to copy the source image before use.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskSizeGB</td>
<td>The size of the disk in gigabytes (GB).</td>
<td>Any size between 16 GB and 65536 GB.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskType</td>
<td>The type of disk.</td>
<td>Either the default <code>pd-ssd</code> or the <code>pd-standard</code> disk type. The control plane nodes must be the <code>pd-ssd</code> disk type. The worker nodes can be either type.</td>
</tr>
</tbody>
</table>

### 4.5.5.2. Sample customized install-config.yaml file for GCP

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.
apiVersion: v1
baseDomain: example.com

controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
    osDisk:
      diskType: pd-ssd
      diskSizeGB: 1024
  replicas: 3

compute:
  - hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
        - us-central1-a
        - us-central1-c
    osDisk:
      diskType: pd-standard
      diskSizeGB: 128
  replicas: 3

metadata:
  name: test-cluster

networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
  platform:
    gcp:
      projectId: openshift-production
      region: us-central1
      pullSecret: '{"auths": ...}'
      fips: false
      sshKey: ssh-ed25519 AAAA...

Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings.
Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

You can optionally provide the `sshKey` value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 4.5.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  
     noProxy: example.com  
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...  
   ```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.
   2. A proxy URL to use for creating HTTPS connections outside the cluster.
   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
   4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the `openshift-config` namespace to hold the additional CA certificates. If you provide **additionalTrustBundle** and at least one proxy setting, the **Proxy** object is configured to reference the **user-ca-bundle** config map in the **trustedCA** field. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges the contents specified for the **trustedCA** parameter with the RHCOS trust bundle. The **additionalTrustBundle** field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

   **NOTE**

   The installation program does not support the proxy **readinessEndpoints** field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 4.5.6. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

**Phase 2**

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

### 4.5.7. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**
Create the `install-config.yaml` file and complete any modifications to it.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   
   where:
   
   `<installation_directory>`
   
   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   
   where:
   
   `<installation_directory>`
   
   Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

   **Specify a different VXLAN port for the OpenShift SDN network provider**

   ```
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     defaultNetwork:
       openshiftSDNConfig:
         vxlanPort: 4800
   ```

4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

**4.5.8. Cluster Network Operator configuration**

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the `Network` API in the `operator.openshift.io` API group.
The CNO configuration inherits the following fields during cluster installation from the Network API in the Network.config.openshift.io API group and these fields cannot be changed:

**clusterNetwork**
- IP address pools from which pod IP addresses are allocated.

**serviceNetwork**
- IP address pool for services.

**defaultNetwork.type**
- Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

### 4.5.8.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always <strong>cluster</strong>.</td>
</tr>
<tr>
<td>spec.clusterNetwork</td>
<td>array</td>
<td>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.32.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the install-config.yaml file.</td>
</tr>
<tr>
<td>spec.serviceNetwork</td>
<td>array</td>
<td>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the install-config.yaml file.</td>
</tr>
<tr>
<td>spec.defaultNetwork</td>
<td>object</td>
<td>Configures the Container Network Interface (CNI) cluster network provider for the cluster network.</td>
</tr>
</tbody>
</table>
The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.

### defaultNetwork object configuration

The values for the `defaultNetwork` object are defined in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>Either <a href="#">OpenShiftSDN</a> or <a href="#">OVNKubernetes</a>. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

**NOTE**

OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshiftSDNConfig</td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td>ovnKubernetesConfig</td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

### Configuration for the OpenShift SDN CNI cluster network provider

The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is <code>NetworkPolicy</code>. The values <code>Multitenant</code> and <code>Subnet</code> are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>
The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.

This value cannot be changed after cluster installation.

The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.

If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.

On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.
The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.

This value cannot be changed after cluster installation.

genevePort

The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.

Example OVN-Kubernetes configuration

defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081

kubeProxyConfig object configuration

The values for the kubeProxyConfig object are defined in the following table:

Table 4.16. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.</td>
</tr>
</tbody>
</table>

NOTE

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxyArguments.iptables-min-sync-period</td>
<td>array</td>
<td>The minimum duration before refreshing <code>iptables</code> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go <code>time</code> package. The default value is: <code>kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</code></td>
</tr>
</tbody>
</table>

### 4.5.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> --log-level=info
   ```

   **1** For `<installation_directory>`, specify the
   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**
NOTE

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

4.5.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of `oc`, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of `oc`.

4.5.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvf <file>

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

4.5.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:

   C:\> oc <command>

4.5.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ``

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

4.5.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   ```
   system:admin
   ```

**Additional resources**

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

4.5.12. Telemetry access for OpenShift Container Platform
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

4.5.13. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

4.6. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.6, you can install a cluster on Google Cloud Platform (GCP) in a restricted network by creating an internal mirror of the installation release content on an existing Google Virtual Private Cloud (VPC).

**IMPORTANT**

You can install an OpenShift Container Platform cluster by using mirrored installation release content, but your cluster will require internet access to use the GCP APIs.

4.6.1. Prerequisites

- You mirrored the images for a disconnected installation to your registry and obtained the imageContentSources data for your version of OpenShift Container Platform.

**IMPORTANT**

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You have an existing VPC in GCP. While installing a cluster in a restricted network that uses installer-provisioned infrastructure, you cannot use the installer-provisioned VPC. You must use a user-provisioned VPC that satisfies one of the following requirements:
  - Contains the mirror registry
  - Has firewall rules or a peering connection to access the mirror registry hosted elsewhere
- You reviewed details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to. While you might need to grant access to more sites, you must grant access to *.googleapis.com and accounts.google.com.
4.6.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

4.6.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The `ClusterVersion` status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

4.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.
4.6.4. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N " \n   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

**Example output**

Agent pid 31874
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>  # 1
   ```

   **Example output**

   ```bash
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   1 Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

4. Set the `GOOGLE_APPLICATION_CREDENTIALS` environment variable to the full path to your service account private key file.

   ```bash
   $ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"
   ```

5. Verify that the credentials were applied.

   ```bash
   $ gcloud auth list
   ```

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**4.6.5. Creating the installation configuration file**

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

- Have the `imageContentSources` values that were generated during mirror registry creation.

- Obtain the contents of the certificate for your mirror registry.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>  # 1
   ```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `gcp` as the platform to target.

iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

v. Select the region to deploy the cluster to.

vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

vii. Enter a descriptive name for your cluster.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Edit the `install-config.yaml` file to provide the additional information that is required for an installation in a restricted network.

a. Update the `pullSecret` value to contain the authentication information for your registry:

```
pullSecret: {"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>"},"email": "you@example.com"}}
```

For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

b. Add the `additionalTrustBundle` parameter and value.

-
additionalTrustBundle: |
-----BEGIN CERTIFICATE-----
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
-----END CERTIFICATE-----

The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

c. Define the network and subnets for the VPC to install the cluster in under the parent platform.gcp field:

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

For platform.gcp.network, specify the name for the existing Google VPC. For platform.gcp.controlPlaneSubnet and platform.gcp.computeSubnet, specify the existing subnets to deploy the control plane machines and compute machines, respectively.

d. Add the image content resources, which look like this excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
    source: quay.example.com/openshift-release-dev/ocp-release
  - mirrors:
    - <mirror_host_name>:5000/<repo_name>/release
      source: registry.example.com/ocp/release
```

To complete these values, use the imageContentSources that you recorded during mirror registry creation.

3. Make any other modifications to the install-config.yaml file that you require. You can find more information about the available parameters in the Installation configuration parameters section.

4. Back up the install-config.yaml file so that you can use it to install multiple clusters.

**IMPORTANT**

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

4.6.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.
NOTE

After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

4.6.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 4.17. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}</code>, <code>{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere.** For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{ &quot;auths&quot;: { &quot;cloud.openshift.com&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; }, &quot;quay.io&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;you@example.com&quot; } } }</code></td>
</tr>
</tbody>
</table>

### 4.6.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 4.18. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machine Network</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
4.6.5.13. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 4.19. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><code>Enabled</code> or <code>Disabled</code></td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td><code>worker</code></td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or {}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;)</td>
</tr>
<tr>
<td></td>
<td>dynamically tries to determine the capabilities of the provided credentials,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with a preference for mint mode on the platforms where multiple modes are</td>
<td></td>
</tr>
<tr>
<td></td>
<td>supported.</td>
<td></td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is **false** (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>fips</strong></td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td><strong>imageContentSources</strong></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><strong>imageContentSources.source</strong></td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td><strong>imageContentSources.mirrors</strong></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td><strong>publish</strong></td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><strong>Internal</strong> or <strong>External</strong>. To deploy a private cluster, which cannot be accessed from the internet, set <strong>publish</strong> to <strong>Internal</strong>. The default value is <strong>External</strong>.</td>
</tr>
</tbody>
</table>
### sshKey

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

- `sshKey: <key1>`
- `<key2>`
- `<key3>`

---

### 4.6.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

**Table 4.20. Additional GCP parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.gcp.network</code></td>
<td>The name of the existing VPC that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.gcp.region</code></td>
<td>The name of the GCP region that hosts your cluster.</td>
<td>Any valid region name, such as <code>us-central1</code></td>
</tr>
<tr>
<td><code>platform.gcp.type</code></td>
<td>The GCP machine type.</td>
<td>The GCP machine type.</td>
</tr>
<tr>
<td><code>platform.gcp.zones</code></td>
<td>The availability zones where the installation program creates machines for the specified MachinePool.</td>
<td>A list of valid GCP availability zones, such as <code>us-central1-a</code>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>platform.gcp.controlPlaneSubnet</code></td>
<td>The name of the existing subnet in your VPC that you want to deploy your control plane machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.gcp.computeSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your compute machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.licenses</td>
<td>A list of license URLs that must be applied to the compute images.</td>
<td>Any license available with the license API, such as the license to enable nested virtualization. You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installer to copy the source image before use.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskSizeGB</td>
<td>The size of the disk in gigabytes (GB).</td>
<td>Any size between 16 GB and 65536 GB.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskType</td>
<td>The type of disk.</td>
<td>Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. The worker nodes can be either type.</td>
</tr>
</tbody>
</table>

### 4.6.5.2. Sample customized install-config.yaml file for GCP

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane: 1
hyperthreading: Enabled 4
name: master
platform:
gcp:
```

---

**platform.gcp.computeSubnet**

The name of the existing subnet in your VPC that you want to deploy your compute machines to.

**platform.gcp.licenses**

A list of license URLs that must be applied to the compute images.

**IMPORTANT**

The `licenses` parameter is a deprecated field and nested virtualization is enabled by default. It is not recommended to use this field.

**platform.gcp.osDisk.diskSizeGB**

The size of the disk in gigabytes (GB).

**platform.gcp.osDisk.diskType**

The type of disk.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.
type: n2-standard-4
disks:
  - us-central1-a
  - us-central1-c
osDisk:
  diskType: pd-ssd
diskSizeGB: 1024
replicas: 3
compute: 5
- hyperthreading: Enabled
name: worker
gcp:
  type: n2-standard-4
disks:
  - us-central1-a
  - us-central1-c
osDisk:
  diskType: pd-standard
diskSizeGB: 128
replicas: 3
metadata:
  name: test-cluster
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
gcp:
  projectId: openshift-production
  region: us-central1
  network: existing_vpc
controlPlaneSubnet: control_plane_subnet
computeSubnet: compute_subnet
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}'}
fips: false
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle:
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
imageContentSources:
  - mirrors:
    - <local_registry>/<local_repository_name>/release
    source: quay.io/openshift-release-dev/ocp-release
  - mirrors:
    - <local_registry>/<local_repository_name>/release
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
189. Required. The installation program prompts you for this value.

25 If you do not provide these parameters and values, the installation program provides the default value.

36 The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

47 Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as n1-standard-8, for your machines if you disable simultaneous multithreading.

11 Specify the name of an existing VPC.

12 Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.

13 Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.

14 For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example, `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

16 You can optionally provide the sshKey value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.
4.6.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
     noProxy: example.com  # 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...
   ```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2. A proxy URL to use for creating HTTPS connections outside the cluster.
   3. A comma-separated list of destination domain names, IP addresses, or other network
If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide

NOTE
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

NOTE
Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

4.6.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:

   - The `GOOGLE_CREDENTIALS`, `GOOGLE_CLOUD_KEYFILE_JSON`, or `G CLOUD_KEYFILE_JSON` environment variables
   - The `~/.gcp/osServiceAccount.json` file
   - The `gcloud cli` default credentials

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \ 1
   --log-level=info 2
   ```
1. For `<installation_directory>`, specify the location of your customized `.install-config.yaml` file.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```plaintext
... INFO Install complete!
INFO To access the cluster as the system:admin user when using `oc`, run `export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig`
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.
3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
   - If you assigned the **Owner** role to your service account, you can remove that role and replace it with the **Viewer** role.
   - If you included the **Service Account Key Admin** role, you can remove it.

### 4.6.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) in order to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of **oc**.

#### 4.6.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Linux Client** entry and save the file.
4. Unpack the archive:
   
   ```bash
   $ tar xvzf <file>
   ```
5. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:
   
   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the **oc** command:

```bash
$ oc <command>
```

#### 4.6.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the **oc** binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

   After you install the OpenShift CLI, it is available using the **oc** command:

   ```
   C:\> oc <command>
   ```

### 4.6.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpick and unzip the archive.

5. Move the **oc** binary to a directory on your **PATH**.
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the **oc** command:

   ```
   $ oc <command>
   ```

### 4.6.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

**Procedure**
1. Export the **kubeadmin** credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   **Example output**

   ```bash
   system:admin
   ```

### 4.6.9. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```bash
  $ oc patch OperatorHub cluster --type json \\
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
  ```

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the **Administration → Cluster Settings → Global Configuration → OperatorHub** page, click the **Sources** tab, where you can create, delete, disable, and enable individual sources.

### 4.6.10. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to **OpenShift Cluster Manager**.

After you confirm that your **OpenShift Cluster Manager** inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, **use subscription watch** to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See **About remote health monitoring** for more information about the Telemetry service

### 4.6.11. Next steps
- Validate an installation.
- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the \texttt{must-gather} tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can opt out of remote health reporting.

4.7. INSTALLING A CLUSTER ON GCP INTO AN EXISTING VPC

In OpenShift Container Platform version 4.6, you can install a cluster into an existing Virtual Private Cloud (VPC) on Google Cloud Platform (GCP). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the \texttt{install-config.yaml} file before you install the cluster.

4.7.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

4.7.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

\textbf{IMPORTANT}

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.
4.7.3. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874
NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name>
```

Example output

```
Identity added: /home/<you>/<path>/<file_name> (computer_name)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

4. Set the `GOOGLE_APPLICATION_CREDENTIALS` environment variable to the full path to your service account private key file.

```
$ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>
```

5. Verify that the credentials were applied.

```
$ gcloud auth list
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

4.7.4. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.
The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ tar xvf openshift-install-linux.tar.gz
```

Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 4.7.5. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.
   
   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
b. At the prompts, provide the configuration details for your cloud:
   
i. Optional: Select an SSH key to use to access your cluster machines.

   **NOTE**
   
   For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

   ii. Select `gcp` as the platform to target.

   iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

   iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

   v. Select the region to deploy the cluster to.

   vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

   vii. Enter a descriptive name for your cluster.

   viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**
   
   The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

4.7.5.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.
NOTE
After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT
The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

4.7.5.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 4.21. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiVersion</code></td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td><code>baseDomain</code></td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td><code>metadata</code></td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td><code>metadata.name</code></td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}</code>, <code>{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere.**

For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

### pullSecret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

#### 4.7.5.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

### Table 4.22. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
### networking.machineNetwork.cidr

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

4.7.5.13. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following “Machine-pool” table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><code>controlPlane.hypertreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hyperthreading</em>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td><strong>IMPORTANT</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td><code>controlPlane.name</code></td>
<td>Required if you use <code>controlPlane</code>. The name of the machine pool.</td>
<td><code>master</code></td>
</tr>
<tr>
<td><code>controlPlane.platform</code></td>
<td>Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or {}</code></td>
</tr>
<tr>
<td><code>controlPlane.replicas</code></td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
</tbody>
</table>

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>imageContentSources</th>
<th>Sources and repositories for the release-image content.</th>
<th>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. To deploy a private cluster, which cannot be accessed from the internet, set <code>publish</code> to <code>Internal</code>. The default value is <code>External</code>.</td>
</tr>
</tbody>
</table>
The SSH key or keys to authenticate access your cluster machines.

**NOTE**
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```plaintext
sshKey:
  <key1>
  <key2>
  <key3>
```

### 4.7.5.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

**Table 4.24. Additional GCP parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.gcp.network</code></td>
<td>The name of the existing VPC that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.gcp.region</code></td>
<td>The name of the GCP region that hosts your cluster.</td>
<td>Any valid region name, such as <code>us-central1</code>.</td>
</tr>
<tr>
<td><code>platform.gcp.type</code></td>
<td>The GCP machine type.</td>
<td>The GCP machine type.</td>
</tr>
<tr>
<td><code>platform.gcp.zones</code></td>
<td>The availability zones where the installation program creates machines for the specified MachinePool.</td>
<td>A list of valid GCP availability zones, such as <code>us-central1-a</code>, in a YAML sequence.</td>
</tr>
<tr>
<td><code>platform.gcp.controlPlaneSubnet</code></td>
<td>The name of the existing subnet in your VPC that you want to deploy your control plane machines to.</td>
<td>The subnet name.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.gcp.computeSubnet</code></td>
<td>The name of the existing subnet in your VPC that you want to deploy your compute machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td><code>platform.gcp.licenses</code></td>
<td>A list of license URLs that must be applied to the compute images.</td>
<td>Any license available with the license API, such as the license to enable nested virtualization. You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installer to copy the source image before use.</td>
</tr>
<tr>
<td><code>platform.gcp.osDisk.diskSizeGB</code></td>
<td>The size of the disk in gigabytes (GB).</td>
<td>Any size between 16 GB and 65536 GB.</td>
</tr>
<tr>
<td><code>platform.gcp.osDisk.diskType</code></td>
<td>The type of disk.</td>
<td>Either the default <code>pd-ssd</code> or the <code>pd-standard</code> disk type. The control plane nodes must be the <code>pd-ssd</code> disk type. The worker nodes can be either type.</td>
</tr>
</tbody>
</table>

### 4.7.5.2. Sample customized install-config.yaml file for GCP

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane:  # 1
hyperthreading: Enabled  # 4
name: master
platform:
gcp:
```
Required. The installation program prompts you for this value.

If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, `-`, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.
Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as **n1-standard-8**, for your machines if you disable simultaneous multithreading.

Specify the name of an existing VPC.

Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.

Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

### 4.7.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

**Prerequisites**

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s **spec.noProxy** field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
...

1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
2 A proxy URL to use for creating HTTPS connections outside the cluster.
3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
4 If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RH COS trust bundle. The additionalTrustBundle field is required unless the proxy's identity certificate is signed by an authority from the RH COS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 4.7.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Remove any existing GCP credentials that do not use the service account key for the GCP account that you configured for your cluster and that are stored in the following locations:
   - The `GOOGLE_CREDENTIALS`, `GOOGLE_CLOUD_KEYFILE.JSON`, or `GCLOUD_KEYFILE.JSON` environment variables
   - The `~/.gcp/osServiceAccount.json` file
   - The `gcloud cli` default credentials

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \\ 1
   --log-level=info 2
   
   1 For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.
   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.
   ```

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.
When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

3. Optional: You can reduce the number of permissions for the service account that you used to install the cluster.
   - If you assigned the `Owner` role to your service account, you can remove that role and replace it with the `Viewer` role.
   - If you included the `Service Account Key Admin` role, you can remove it.

### 4.7.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

4.7.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

4.7.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the oc command:
4.7.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

4.7.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami
4.7.9. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

4.8. INSTALLING A PRIVATE CLUSTER ON GCP

In OpenShift Container Platform version 4.6, you can install a private cluster into an existing VPC on Google Cloud Platform (GCP). The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

4.8.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.

- Configure a GCP account to host the cluster.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

4.8.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the Internet.
By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

To deploy a private cluster, you must use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.

Additionally, you must deploy a private cluster from a machine that has access the API services for the cloud you provision to, the hosts on the network that you provision, and to the internet to obtain installation media. You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

### 4.8.2.1. Private clusters in GCP

To create a private cluster on Google Cloud Platform (GCP), you must provide an existing private VPC and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

The cluster still requires access to Internet to access the GCP APIs.

The following items are not required or created when you install a private cluster:

- Public subnets
- Public network load balancers, which support public ingress
- A public DNS zone that matches the `baseDomain` for the cluster

The installation program does use the `baseDomain` that you specify to create a private DNS zone and the required records for the cluster. The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

Because it is not possible to limit access to external load balancers based on source tags, the private cluster uses only internal load balancers to allow access to internal instances.

The internal load balancer relies on instance groups rather than the target pools that the network load balancers use. The installation program creates instance groups for each zone, even if there is no instance in that group.

- The cluster IP address is internal only.
- One forwarding rule manages both the Kubernetes API and machine config server ports.
- The backend service is comprised of each zone’s instance group and, while it exists, the bootstrap instance group.
- The firewall uses a single rule that is based on only internal source ranges.

#### 4.8.2.1.1. Limitations

No health check for the Machine config server, `/healthz`, runs because of a difference in load balancer functionality. Two internal load balancers cannot share a single IP address, but two network load balancers can share a single external IP address. Instead, the health of an instance is determined entirely by the `/readyz` check on port 6443.
4.8.3. About using a custom VPC

In OpenShift Container Platform 4.6, you can deploy a cluster into an existing VPC in Google Cloud Platform (GCP). If you do, you must also use existing subnets within the VPC and routing rules.

By deploying OpenShift Container Platform into an existing GCP VPC, you might be able to avoid limit constraints in new accounts or more easily abide by the operational constraints that your company’s guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VPC yourself.

4.8.3.1. Requirements for using your VPC

The installation program will no longer create the following components:

- VPC
- Subnets
- Cloud router
- Cloud NAT
- NAT IP addresses

If you use a custom VPC, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VPC options like DHCP, so you must do so before you install the cluster.

Your VPC and subnets must meet the following characteristics:

- The VPC must be in the same GCP project that you deploy the OpenShift Container Platform cluster to.
- To allow access to the Internet from the control plane and compute machines, you must configure cloud NAT on the subnets to allow egress to it. These machines do not have a public address. Even if you do not require access to the Internet, you must allow egress to the VPC network to obtain the installation program and images. Because multiple cloud NATs cannot be configured on the shared subnets, the installation program cannot configure it.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the subnets that you specify exist and belong to the VPC that you specified.
- The subnet CIDRs belong to the machine CIDR.
- You must provide a subnet to deploy the cluster control plane and compute machines to. You can use the same subnet for both machine types.

If you destroy a cluster that uses an existing VPC, the VPC is not deleted.

4.8.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different
resources in your clouds than others. For example, you might be able to create application-specific items, like instances, buckets, and load balancers, but not networking-related components such as VPCs, subnets, or Ingress rules.

The GCP credentials that you use when you create your cluster do not need the networking permissions that are required to make VPCs and core networking components within the VPC, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage, and nodes.

4.8.3.3. Isolation between clusters

If you deploy OpenShift Container Platform to an existing network, the isolation of cluster services is preserved by firewall rules that reference the machines in your cluster by the cluster’s infrastructure ID. Only traffic within the cluster is allowed.

If you deploy multiple clusters to the same VPC, the following components might share access between clusters:

- The API, which is globally available with an external publishing strategy or available throughout the network in an internal publishing strategy
- Debugging tools, such as ports on VM instances that are open to the machine CIDR for SSH and ICMP access

4.8.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.8.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.
NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" \\
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874

   NOTE

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```bash
   $ ssh-add <path>/<file_name>
   ```
Example output

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

4. Set the GOOGLE_APPLICATION_CREDENTIALS environment variable to the full path to your service account private key file.

   $ export GOOGLE_APPLICATION_CREDENTIALS="<your_service_account_file>"

5. Verify that the credentials were applied.

   $ gcloud auth list

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

4.8.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.
4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

4.8.7. Manually creating the installation configuration file

For installations of a private OpenShift Container Platform cluster that are only accessible from an internal network and are not visible to the Internet, you must manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```bash
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

**NOTE**

You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

4.8.7.1. Installation configuration parameters
Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 4.8.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 4.25. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
</tbody>
</table>
The name of the cluster. DNS records for the cluster are all subdomains of 
{{.metadata.name}}. {{.baseDomain}}.

String of lowercase letters, hyphens (-), and periods (.), such as dev.

The configuration for the specific platform upon which to perform the installation: 
aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.<platform> parameters, consult the following table for your specific platform.

Object

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```
{
    "auths":{
        "cloud.openshift.com":{
            "auth":"b3Blb=",
            "email":"you@example.com"
        },
        "quay.io":{
            "auth":"b3Blb=",
            "email":"you@example.com"
        }
    }
}
```

4.8.7.12. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 4.26. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.26. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
</tbody>
</table>

808
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td>networking.networkType</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <strong>10.128.0.0/14</strong> with a host prefix of <strong>/23</strong>.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: <strong>10.128.0.0/14</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: <strong>23</strong></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking.clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cidr: <strong>10.128.0.0/14</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: <strong>23</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to <strong>23</strong> then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of <strong>23</strong> provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is <strong>23</strong>.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <strong>172.30.0.0/16</strong>.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>172.30.0.0/16</strong></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The</td>
<td>An IP network block in CIDR notation.</td>
</tr>
<tr>
<td></td>
<td>default value is 10.0.0.0/16 for all platforms other than libvirt. For</td>
<td>For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>libvirt, the default value is 192.168.126.0/24.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NIC resides in.</td>
</tr>
</tbody>
</table>

### 4.8.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 4.27. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>certificate store. This trust bundle may also be used when a proxy has</td>
<td></td>
</tr>
<tr>
<td></td>
<td>been configured.</td>
<td></td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Currently, heterogeneous clusters are not supported, so all pools must</td>
<td></td>
</tr>
<tr>
<td></td>
<td>specify the same architecture. Valid values are amd64 (the default).</td>
<td></td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or *hyperthreading*, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hyperthreading</em>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>imageContentSources</strong></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><strong>imageContentSources.source</strong></td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td><strong>imageContentSources.mirrors</strong></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
4.8.7.1.4. Additional Google Cloud Platform (GCP) configuration parameters

Additional GCP configuration parameters are described in the following table:

Table 4.28. Additional GCP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.gcp.network</code></td>
<td>The name of the existing VPC that you want to deploy your cluster to.</td>
<td>String.</td>
</tr>
<tr>
<td><code>platform.gcp.region</code></td>
<td>The name of the GCP region that hosts your cluster.</td>
<td>Any valid region name, such as <code>us-central1</code>.</td>
</tr>
<tr>
<td><code>platform.gcp.type</code></td>
<td>The GCP machine type.</td>
<td>The GCP machine type.</td>
</tr>
<tr>
<td><code>platform.gcp.zones</code></td>
<td>The availability zones where the installation program creates machines for the specified MachinePool.</td>
<td>A list of valid GCP availability zones, such as <code>us-central1-a</code>, in a YAML sequence.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.gcp.controlPlaneSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your control plane machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.computeSubnet</td>
<td>The name of the existing subnet in your VPC that you want to deploy your compute machines to.</td>
<td>The subnet name.</td>
</tr>
<tr>
<td>platform.gcp.licenses</td>
<td>A list of license URLs that must be applied to the compute images.</td>
<td>Any license available with the license API, such as the license to enable nested virtualization. You cannot use this parameter with a mechanism that generates pre-built images. Using a license URL forces the installer to copy the source image before use.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskSizeGB</td>
<td>The size of the disk in gigabytes (GB).</td>
<td>Any size between 16 GB and 65536 GB.</td>
</tr>
<tr>
<td>platform.gcp.osDisk.diskType</td>
<td>The type of disk.</td>
<td>Either the default pd-ssd or the pd-standard disk type. The control plane nodes must be the pd-ssd disk type. The worker nodes can be either type.</td>
</tr>
</tbody>
</table>

### 4.8.7.2. Sample customized install-config.yaml file for GCP

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

**IMPORTANT**

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.
Required. The installation program prompts you for this value.
If you do not provide these parameters and values, the installation program provides the default value.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as n1-standard-8, for your machines if you disable simultaneous multithreading.

Specify the name of an existing VPC.

Specify the name of the existing subnet to deploy the control plane machines to. The subnet must belong to the VPC that you specified.

Specify the name of the existing subnet to deploy the compute machines to. The subnet must belong to the VPC that you specified.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

You can optionally provide the sshKey value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

How to publish the user-facing endpoints of your cluster. Set publish to Internal to deploy a private cluster, which cannot be accessed from the Internet. The default value is External.

4.8.7.3. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>
     httpsProxy: https://<username>:<pswd>@<ip>:<port>
     noProxy: example.com
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ... 
   ```

   - A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`
   - A proxy URL to use for creating HTTPS connections outside the cluster.
   - A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
   - If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents...
specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

4.8.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   $ ./openshift-install create cluster --dir <installation_directory> \ 1
   --log-level=info  2

   1 For <installation_directory>, specify the
   2 To view different installation details, specify warn, debug, or error instead of info.
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

Example output

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=~/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

NOTE

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

4.8.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

4.8.9.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xzvf <file>

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

4.8.9.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.

   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
4.8.9.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

4.8.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```
Example output

```
  system:admin
```

Additional resources

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

### 4.8.11. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

### 4.8.12. Next steps

- Customize your cluster.
- If necessary, you can [opt out of remote health reporting](#).

### 4.9. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE IN GCP BY USING DEPLOYMENT MANAGER TEMPLATES

In OpenShift Container Platform version 4.6, you can install a cluster on Google Cloud Platform (GCP) that uses infrastructure that you provide.

The steps for performing a user-provided infrastructure install are outlined here. Several [Deployment Manager](#) templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.

**IMPORTANT**

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

### 4.9.1. Prerequisites

- Review details about the [OpenShift Container Platform installation and update](#) processes.
• If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

• If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

  NOTE
  Be sure to also review this site list if you are configuring a proxy.

4.9.2. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The kube-controller-manager only approves the kubelet client CSRs. The machine-approver cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

4.9.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

• Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

• Access Quay.io to obtain the packages that are required to install your cluster.

• Obtain the packages that are required to perform cluster updates.

  IMPORTANT
  If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.9.4. Configuring your GCP project

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

4.9.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.
Procedure

- Create a project to host your OpenShift Container Platform cluster. See Creating and Managing Projects in the GCP documentation.

**IMPORTANT**

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the `api-int.<cluster_name>.<base_domain>` URL; the Premium Tier is required for internal load balancing.

4.9.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

**Prerequisites**

- You created a project to host your cluster.

**Procedure**

- Enable the following required API services in the project that hosts your cluster. See Enabling services in the GCP documentation.

**Table 4.29. Required API services**

<table>
<thead>
<tr>
<th>API service</th>
<th>Console service name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Deployment Manager V2 API</td>
<td>deploymentmanager.googleapis.com</td>
</tr>
<tr>
<td>Compute Engine API</td>
<td>compute.googleapis.com</td>
</tr>
<tr>
<td>Google Cloud APIs</td>
<td>cloudapis.googleapis.com</td>
</tr>
<tr>
<td>Cloud Resource Manager API</td>
<td>cloudresourcemanager.googleapis.com</td>
</tr>
<tr>
<td>Google DNS API</td>
<td>dns.googleapis.com</td>
</tr>
<tr>
<td>IAM Service Account Credentials API</td>
<td>iamcredentials.googleapis.com</td>
</tr>
<tr>
<td>Identity and Access Management (IAM) API</td>
<td>iam.googleapis.com</td>
</tr>
<tr>
<td>Service Management API</td>
<td>servicemanagement.googleapis.com</td>
</tr>
<tr>
<td>Service Usage API</td>
<td>serviceusage.googleapis.com</td>
</tr>
<tr>
<td>Google Cloud Storage JSON API</td>
<td>storage-api.googleapis.com</td>
</tr>
</tbody>
</table>
4.9.4.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.

   **NOTE**
   If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see Google Domains.

2. Create a public hosted zone for your domain or subdomain in your GCP project. See Creating public zones in the GCP documentation.
   Use an appropriate root domain, such as openshiftcorp.com, or subdomain, such as clusters.openshiftcorp.com.

3. Extract the new authoritative name servers from the hosted zone records. See Look up your Cloud DNS name servers in the GCP documentation.
   You typically have four name servers.

4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: How to switch to custom name servers.

5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See Migrating to Cloud DNS in the GCP documentation.

6. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain. This process might include a request to your company’s IT department or the division that controls the root domain and DNS services for your company.

### 4.9.4.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default Quotas do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

<table>
<thead>
<tr>
<th>API service</th>
<th>Console service name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Storage</td>
<td>storage-component.googleapis.com</td>
</tr>
</tbody>
</table>

Table 4.30. GCP resources used in a default cluster
<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Location</th>
<th>Total resources required</th>
<th>Resources removed after bootstrap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service account</td>
<td>IAM</td>
<td>Global</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Firewall rules</td>
<td>Networking</td>
<td>Global</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>Forwarding rules</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Health checks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Images</td>
<td>Compute</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Networks</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routers</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routes</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Subnetworks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Target pools</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

**NOTE**

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- asia-east2
- asia-northeast2
- asia-south1
- australia-southeast1
- europe-north1
- europe-west2
- europe-west3
- europe-west6
You can increase resource quotas from the GCP console, but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

### 4.9.4.5. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

**Prerequisites**

- You created a project to host your cluster.

**Procedure**

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See **Creating a service account** in the GCP documentation.

2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See **Granting roles to a service account for specific resources**.

   **NOTE**

   While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. Create the service account key in JSON format. See **Creating service account keys** in the GCP documentation.

   The service account key is required to create a cluster.

### 4.9.4.5.1. Required GCP permissions

When you attach the **Owner** role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. To deploy an OpenShift Container Platform cluster, the service account requires the following permissions. If you deploy your cluster into an existing VPC, the service account does not require certain networking permissions, which are noted in the following lists:

**Required roles for the installation program**

- Compute Admin
- Security Admin
- Service Account Admin
Service Account User

Storage Admin

Required roles for creating network resources during installation

- DNS Administrator

Required roles for user-provisioned GCP infrastructure

- Deployment Manager Editor
- Service Account Key Admin

Optional roles

For the cluster to create new limited credentials for its Operators, add the following role:

- Service Account Key Admin

The roles are applied to the service accounts that the control plane and compute machines use:

<table>
<thead>
<tr>
<th>Account</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Plane</td>
<td>roles/compute.instanceAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.networkAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.securityAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
<tr>
<td></td>
<td>roles/iam.serviceAccountUser</td>
</tr>
<tr>
<td>Compute</td>
<td>roles/compute.viewer</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
</tbody>
</table>

4.9.4.6. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
4.9.4.7. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

Prerequisites

- You created a project to host your cluster.
- You created a service account and granted it the required permissions.

Procedure

1. Install the following binaries in $PATH:
   - gcloud
4.9.5. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the `install-config.yaml` file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate `var` partition during the preparation phases of installation.

4.9.5.1. Optional: Creating a separate `/var` partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

**IMPORTANT**

If you follow the steps to create a separate `/var` partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:
$ openshift-install create manifests --dir $HOME/clusterconfig

Example output

? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift

3. Optional: Confirm that the installation program created manifests in the
clusterconfig/openshift directory:

$ ls $HOME/clusterconfig/openshift/

Example output

99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...

4. Create a MachineConfig object and add it to a file in the openshift directory. For example,
name the file 98-var-partition.yaml, change the disk device name to the name of the storage
device on the worker systems, and set the storage size as appropriate. This example places the
/var directory on a separate partition:

apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    - machineconfiguration.openshift.io/role: worker
name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
  storage:
    disks:
    - device: /dev/<device_name>  
      partitions:
        - label: var
          startMiB: <partition_start_offset>  
          sizeMiB: <partition_size>  
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
          systemd:
            units:
              - name: var.mount  
                enabled: true
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the \texttt{Where=} directive. For example, for a filesystem mounted on \texttt{/var/lib/containers}, the unit must be named \texttt{var-lib-containers.mount}.

The \texttt{prjquota} mount option must be enabled for filesystems used for container storage.

\begin{verbatim}
[Unit]
Before=local-fs.target

[Mount]
What=/dev/disk/by-partlabel/var
Where=/var
Options=defaults,prjquota

[Install]
WantedBy=local-fs.target
\end{verbatim}

\textbf{NOTE}

When creating a separate \texttt{/var} partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Run \texttt{openshift-install} again to create Ignition configs from a set of files in the \texttt{manifest} and \texttt{openshift} subdirectories:

\begin{verbatim}
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
\end{verbatim}

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

\subsection*{4.9.5.2. Creating the installation configuration file}

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

\textbf{Prerequisites}

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

\textbf{Procedure}
1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>  
```

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   IMPORTANT

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

   i. Optional: Select an SSH key to use to access your cluster machines.

   ii. Select `gcp` as the platform to target.

   iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

   iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

   v. Select the region to deploy the cluster to.

   vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

   vii. Enter a descriptive name for your cluster.

   viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

   c. Optional: If you do not want the cluster to provision compute machines, empty the compute pool by editing the resulting `install-config.yaml` file to set `replicas` to 0 for the `compute` pool:

```
compute:
- hyperthreading: Enabled
```
1. Set to 0.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 4.9.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
  httpsProxy: https://<username>:<pswd>@<ip>:<port>
  noProxy: example.com
additionalTrustBundle: |
```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

4.9.5.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the install-config.yaml installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

Example output

```
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift
```

For <installation_directory>, specify the installation directory that contains the install-config.yaml file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Optional: If you do not want the cluster to provision compute machines, remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.
4. Check that the `mastersSchedulable` parameter in the
   `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest
   file is set to `false`. This setting prevents pods from being scheduled on the control plane
   machines:
   
a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   
b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   
c. Save and exit the file.

5. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove
   the `privateZone` and `publicZone` sections from the
   `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:
   
   ```yaml
   apiVersion: config.openshift.io/v1
   kind: DNS
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     baseDomain: example.openshift.com
     privateZone: 1
     id: mycluster-100419-private-zone
     publicZone: 2
     id: example.openshift.com
   status: {}
   ``

   Remove this section completely.

   If you do so, you must add ingress DNS records manually in a later step.

6. To create the Ignition configuration files, run the following command from the directory that
   contains the installation program:
   
   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:
   
   ```sh
   ├── auth
   │   └── kubeadm-password
   │       └── kubectlconfig
   │           └── bootstrap.ign
   │               └── master.ign
   │                   └── metadata.json
   │                       └── worker.ign
   ```

   Additional resources
   
   - Optional: Adding the ingress DNS records
4.9.6. Exporting common variables

4.9.6.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the jq package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  $ jq -r .infraID <installation_directory>/metadata.json

  For <installation_directory>, specify the path to the directory that you stored the installation files in.

Example output

  openshift-vw9j6

  The output of this command is your cluster name and a random string.

4.9.6.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).

NOTE

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Generate the Ignition config files for your cluster.
• Install the jq package.

Procedure

1. Export the following common variables to be used by the provided Deployment Manager templates:

   $ export BASE_DOMAIN='<base_domain>'
   $ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
   $ export NETWORK_CIDR='10.0.0.0/16'
   $ export MASTER_SUBNET_CIDR='10.0.0.0/19'
   $ export WORKER_SUBNET_CIDR='10.0.32.0/19'
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   $ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
   $ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
   $ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
   $ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

4.9.7. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.

NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

• Configure a GCP account.

• Generate the Ignition config files for your cluster.

Procedure

1. Copy the template from the Deployment Manager template for the VPC section of this topic and save it as 01_vpc.py on your computer. This template describes the VPC that your cluster requires.

2. Create a 01_vpc.yaml resource definition file:

   $ cat <<EOF >01_vpc.yaml
   imports:
     - path: 01_vpc.py
   EOF
infrastructure name from the extraction step.

region is the region to deploy the cluster into, for example us-central1.

master_subnet_cidr is the CIDR for the master subnet, for example 10.0.0.0/19.

worker_subnet_cidr is the CIDR for the worker subnet, for example 10.0.32.0/19.

3. Create the deployment by using the gcloud CLI:

```
$ gcloud deployment-manager deployments create $(INFRA_ID)-vpc --config 01_vpc.yaml
```

4.9.7.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

Example 4.1. 01_vpc.py Deployment Manager template

def GenerateConfig(context):

    resources = [
        {'name': context.properties['infra_id'] + '-network',
         'type': 'compute.v1.network',
         'properties': {
             'region': context.properties['region'],
             'autoCreateSubnetworks': False
         }},
        {'name': context.properties['infra_id'] + '-master-subnet',
         'type': 'compute.v1.subnetwork',
         'properties': {
             'region': context.properties['region'],
             'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
             'ipCidrRange': context.properties['master_subnet_cidr']
         }},
        {'name': context.properties['infra_id'] + '-worker-subnet',
         'type': 'compute.v1.subnetwork',
         'properties': {
             'region': context.properties['region'],
             'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
             'ipCidrRange': context.properties['worker_subnet_cidr']
         }
```
4.9.8. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 4.32. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
</tbody>
</table>
**UDP**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 4.33. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 4.34. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements

The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers

Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer.
Configure the following ports on both the front and back of the load balancers:

### Table 4.35. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

### Table 4.36. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>
**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

### 4.9.9. Creating load balancers in GCP

You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

1. Copy the template from the Deployment Manager template for the internal load balancer section of this topic and save it as `02_lb_int.py` on your computer. This template describes the internal load balancing objects that your cluster requires.

2. For an external cluster, also copy the template from the Deployment Manager template for the external load balancer section of this topic and save it as `02_lb_ext.py` on your computer. This template describes the external load balancing objects that your cluster requires.

3. Export the variables that the deployment template uses:
   
   a. Export the cluster network location:
      
      ```bash
      $ export CLUSTER_NETWORK=('gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink')
      ```
   
   b. Export the control plane subnet location:
      
      ```bash
      $ export CONTROL_SUBNET=('gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink')
      ```
   
   c. Export the three zones that the cluster uses:
      
      ```bash
      ```
4. Create a **02_infra.yaml** resource definition file:

```bash
$ cat <<EOF >02_infra.yaml
imports:
  - path: 02_lb_ext.py
  - path: 02_lb_int.py
resources:
  - name: cluster-lb-ext
    type: 02_lb_ext.py
    properties:
      infra_id: '${INFRA_ID}'
      region: '${REGION}'
  - name: cluster-lb-int
    type: 02_lb_int.py
    properties:
      cluster_network: '${CLUSTER_NETWORK}'
      control_subnet: '${CONTROL_SUBNET}'
      infra_id: '${INFRA_ID}'
      region: '${REGION}'
      zones: 
      - '${ZONE_0}'
      - '${ZONE_1}'
      - '${ZONE_2}'
EOF
```

**1** Required only when deploying an external cluster.

**2** `infra_id` is the **INFRA_ID** infrastructure name from the extraction step.

**3** `region` is the region to deploy the cluster into, for example **us-central1**.

**4** `control_subnet` is the URI to the control subnet.

**5** `zones` are the zones to deploy the control plane instances into, like **us-east1-b, us-east1-c**, and **us-east1-d**.

5. Create the deployment by using the **gcloud** CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:

```bash
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "" -f9`)  
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "" -f9`)  
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "" -f9`)  
```
$ export CLUSTER_IP=('gcloud compute addresses describe ${INFRA_ID}-cluster-ip --region=${REGION} --format json | jq -r .address')

7. For an external cluster, also export the cluster public IP address:

$ export CLUSTER_PUBLIC_IP=('gcloud compute addresses describe ${INFRA_ID}-cluster-public-ip --region=${REGION} --format json | jq -r .address')

4.9.9.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

Example 4.2. 02_lb_ext.py Deployment Manager template

```python
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink]',
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink',
            'target': '$ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```
4.9.9.2. Deployment Manager template for the internal load balancer

You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

Example 4.3. 02_lb_int.py Deployment Manager template

```python
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-instance-group' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
        # probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$ref.' + context.properties['infra_id'] + '-api-internal-health-check.selfLink]",'nloadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
        }
    }]
```

CHAPTER 4. INSTALLING ON GCP

849
You will need this template in addition to the 02_lb_ext.py template when you create an external cluster.

4.9.10. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.

NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure
1. Copy the template from the Deployment Manager template for the private DNS section of this topic and save it as `02_dns.py` on your computer. This template describes the private DNS objects that your cluster requires.

2. Create a `02_dns.yaml` resource definition file:

   ```yaml
   $ cat <<EOF >02_dns.yaml
   imports:
   - path: 02_dns.py
   
   resources:
   - name: cluster-dns
     type: 02_dns.py
     properties:
       infra_id: '${INFRA_ID}'
       cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}'
       cluster_network: '${CLUSTER_NETWORK}'
   EOF
   
1. `infra_id` is the INFRA_ID infrastructure name from the extraction step.

2. `cluster_domain` is the domain for the cluster, for example `openshift.example.com`.

3. `cluster_network` is the selfLink URL to the cluster network.

3. Create the deployment by using the `gcloud` CLI:

   ```shell
   $ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
   ```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

   a. Add the internal DNS entries:

   ```bash
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
   ```

   b. For an external cluster, also add the external DNS entries:

   ```bash
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
   ```
4.9.10.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

```
Example 4.4. 02_dns.py Deployment Manager template

def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-private-zone',
         'type': 'dns.v1.managedZone',
         'properties': {
             'description': '',
             'dnsName': context.properties['cluster_domain'] + '.',
             'visibility': 'private',
             'privateVisibilityConfig': {
                 'networks': [{
                     'networkUrl': context.properties['cluster_network']
                 }]
             }
        }
    ]
    return {'resources': resources}
```

4.9.11. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

1. Copy the template from the Deployment Manager template for firewall rules section of this topic and save it as 03_firewall.py on your computer. This template describes the security groups that your cluster requires.
2. Create a 03_firewall.yaml resource definition file:

```
$ cat <<EOF >03_firewall.yaml
imports:
  - path: 03_firewall.py

resources:
  - name: cluster-firewall
    type: 03_firewall.py
    properties:
      allowed_external_cidr: '0.0.0.0/0'
      infra_id: '${INFRA_ID}'
      cluster_network: '${CLUSTER_NETWORK}'}
      network_cidr: '${NETWORK_CIDR}'}
EOF
```

1. **allowed_external_cidr** is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to `[NETWORK_CIDR]`.

2. **infra_id** is the `INFRA_ID` infrastructure name from the extraction step.

3. **cluster_network** is the selfLink URL to the cluster network.

4. **network_cidr** is the CIDR of the VPC network, for example `10.0.0.0/16`.

3. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config 03_firewall.yaml
```

4.9.11.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

**Example 4.5. 03_firewall.py Deployment Manager template**

```python
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    }, {
        'name': context.properties['infra_id'] + '-api',
```
"type": 'compute.v1.firewall',
'properties': {
  'network': context.properties['cluster_network'],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['6443']
  }],
  'sourceRanges': [context.properties['allowed_external_cidr']],
  'targetTags': [context.properties['infra_id'] + '-master']
},

{name}: context.properties['infra_id'] + '-health-checks',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties['cluster_network'],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['6080', '6443', '22624']
  }],
  'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
  'targetTags': [context.properties['infra_id'] + '-master']
},

{name}: context.properties['infra_id'] + '-etcd',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties['cluster_network'],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['2379-2380']
  }],
  'sourceTags': [context.properties['infra_id'] + '-master'],
  'targetTags': [context.properties['infra_id'] + '-master']
},

{name}: context.properties['infra_id'] + '-control-plane',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties['cluster_network'],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['10257']
  }],
  'IPProtocol': 'tcp',
  'ports': ['10259']
},

{name}: context.properties['infra_id'] + '-control-plane',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties['cluster_network'],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['22623']
  }],
  'sourceTags': [
    context.properties['infra_id'] + '-master',
    context.properties['infra_id'] + '-worker'
  ],
  'targetTags': [context.properties['infra_id'] + '-master']
}
4.9.12. Creating IAM roles in GCP

```json
{name: context.properties[infra_id] + '-internal-network',
'type': 'compute.v1.firewall",
'properties': {
  'network': context.properties[cluster_network],
  'allowed': [{
    'IPProtocol': 'icmp'
  },{
    'IPProtocol': 'tcp',
    'ports': ['22']
  }],
  'sourceRanges': [context.properties[network_cidr]],
  'targetTags': [
    context.properties[infra_id] + '-master',
    context.properties[infra_id] + '-worker'
  ]
}
}, {
  'name': context.properties[infra_id] + '-internal-cluster',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties[cluster_network],
    'allowed': [{
      'IPProtocol': 'udp',
      'ports': ['4789', '6081']
    },{
      'IPProtocol': 'tcp',
      'ports': ['9000-9999']
    },{
      'IPProtocol': 'udp',
      'ports': ['9000-9999']
    },{
      'IPProtocol': 'tcp',
      'ports': ['10250']
    },{
      'IPProtocol': 'tcp',
      'ports': ['30000-32767']
    },{
      'IPProtocol': 'udp',
      'ports': ['30000-32767']
    }],
    'sourceTags': [
      context.properties[infra_id] + '-master',
      context.properties[infra_id] + '-worker'
    ],
    'targetTags': [
      context.properties[infra_id] + '-master',
      context.properties[infra_id] + '-worker'
    ]
  }
}
}

return {'resources': resources}
```
You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

1. Copy the template from the Deployment Manager template for IAM roles section of this topic and save it as `03_iam.py` on your computer. This template describes the IAM roles that your cluster requires.

2. Create a `03_iam.yaml` resource definition file:

   ```
   $ cat <<EOF >03_iam.yaml
   imports:
   - path: 03_iam.py
   resources:
   - name: cluster-iam
     type: 03_iam.py
   properties:
     infra_id: '${INFRA_ID}'
   EOF
   
   1 infra_id is the INFRA_ID infrastructure name from the extraction step.
   ```

3. Create the deployment by using the `gcloud` CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
   ```

4. Export the variable for the master service account:

   ```
   $ export MASTER_SERVICE_ACCOUNT=('gcloud iam service-accounts list --filter "email~^${INFRA_ID}-m@${{PROJECT_NAME}}." --format json | jq -r '.[0].email')
   ```

5. Export the variable for the worker service account:

   ```
   $ export WORKER_SERVICE_ACCOUNT=('gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${{PROJECT_NAME}}." --format json | jq -r '.[0].email')
   ```
6. Export the variable for the subnet that hosts the compute machines:

```bash
$ export COMPUTE_SUBNET=('gcloud compute networks subnets describe ${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink')
```

7. The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```bash
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. Create a service account key and store it locally for later use:

```bash
$ gcloud iam service-accounts keys create service-account-key.json --iam-account=${MASTER_SERVICE_ACCOUNT}
```

### 4.9.12.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

```python
def GenerateConfig(context):
    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }
```
4.9.13. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.

**Procedure**

1. Obtain the RHCOS image from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

   The file name contains the OpenShift Container Platform version number in the format `rhcos-<version>-<arch>-gcp.<arch>.tar.gz`.

2. Create the Google storage bucket:

   ```
   $ gsutil mb gs://<bucket_name>
   ```

3. Upload the RHCOS image to the Google storage bucket:

   ```
   $ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz gs://<bucket_name>
   ```

4. Export the uploaded RHCOS image location as a variable:

   ```
   $ export IMAGE_SOURCE=`gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz`
   ```

5. Create the cluster image:

   ```
   $ gcloud compute images create "${INFRA_ID}-rhcos-image" \ 
     --source-uri="${IMAGE_SOURCE}"
   ```

4.9.14. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.
NOTE

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Ensure pyOpenSSL is installed.

Procedure

1. Copy the template from the Deployment Manager template for the bootstrap machine section of this topic and save it as 04_bootstrap.py on your computer. This template describes the bootstrap machine that your cluster requires.

2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:

   ```
   $ export CLUSTER_IMAGE='gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink'
   ```

3. Create a bucket and upload the bootstrap.ign file:

   ```
   $ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
   $ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
   ```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:

   ```
   $ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
   ```

5. Create a 04_bootstrap.yaml resource definition file:

   ```
   $ cat <<EOF >04_bootstrap.yaml
   imports:
     - path: 04_bootstrap.py
   resources:
     - name: cluster-bootstrap
       type: 04_bootstrap.py
       properties:
         infra_id: '${INFRA_ID}'
   EOF
   ```
infra_id is the INFRA_ID infrastructure name from the extraction step.

region is the region to deploy the cluster into, for example us-central1.

zone is the zone to deploy the bootstrap instance into, for example us-central1-b.

cluster_network is the selfLink URL to the cluster network.

control_subnet is the selfLink URL to the control subnet.

image is the selfLink URL to the RHCOS image.

machine_type is the machine type of the instance, for example n1-standard-4.

root_volume_size is the boot disk size for the bootstrap machine.

bootstrap_ign is the URL output when creating a signed URL.

6. Create the deployment by using the gcloud CLI:

   $ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config 04_bootstrap.yaml

7. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the bootstrap machine manually.

   a. Add the bootstrap instance to the internal load balancer instance group:

      $ gcloud compute instance-groups unmanaged add-instances
      ${INFRA_ID}-bootstrap-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap

   b. Add the bootstrap instance group to the internal load balancer backend service:

      $ gcloud compute backend-services add-backend
      ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}

4.9.14.1. Deployment Manager template for the bootstrap machine
You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 4.7. 04_bootstrap.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-bootstrap-public-ip',
         'type': 'compute.v1.address',
         'properties': {
             'region': context.properties['region']
         }},
        {'name': context.properties['infra_id'] + '-bootstrap',
         'type': 'compute.v1.instance',
         'properties': {
             'disks': [{
                 'autoDelete': True,
                 'boot': True,
                 'initializeParams': {
                     'diskSizeGb': context.properties['root_volume_size'],
                     'sourceImage': context.properties['image']
                 }
             }],
             'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' + context.properties['machine_type'],
             'metadata': {
                 'items': [{
                     'key': 'user-data',
                     'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign'] + '"}},"version":"3.1.0"}}
                 }],
                 'networkInterfaces': [{
                     'subnetwork': context.properties['control_subnet'],
                     'accessConfigs': [{
                         'natIP': '${ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address}'
                     }],
                     'tags': {
                         'items': [
                             context.properties['infra_id'] + '-master',
                             context.properties['infra_id'] + '-bootstrap'
                         ]
                     },
                     'zone': context.properties['zone']
                 }],
                 'namedPorts': [
                     {'name': 'ignition',
                      'port': 22623
```
4.9.15. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

**Procedure**

1. Copy the template from the Deployment Manager template for control plane machines section of this topic and save it as **05_control_plane.py** on your computer. This template describes the control plane machines that your cluster requires.

2. Export the following variable required by the resource definition:

   ```
   $ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
   ```

3. Create a **05_control_plane.yaml** resource definition file:

   ```
   $ cat <<EOF >05_control_plane.yaml
   imports:
   - path: 05_control_plane.py
   ```
infra_id is the INFRA_ID infrastructure name from the extraction step.

zones are the zones to deploy the control plane instances into, for example us-central1-a, us-central1-b, and us-central1-c.

control_subnet is the selfLink URL to the control subnet.

image is the selfLink URL to the RHCOS image.

machine_type is the machine type of the instance, for example n1-standard-4.

service_account_email is the email address for the master service account that you created.

ignition is the contents of the master.ign file.

4. Create the deployment by using the gcloud CLI:

   $ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config 05_control_plane.yaml

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

   - Run the following commands to add the control plane machines to the appropriate instance groups:

     $ gcloud compute instance-groups unmanaged add-instances $INFRA_ID-master-$ZONE_0-instance-group --zone=$ZONE_0 --instances=$INFRA_ID-master-0
     $ gcloud compute instance-groups unmanaged add-instances $INFRA_ID-master-$ZONE_1-instance-group --zone=$ZONE_1 --instances=$INFRA_ID-master-1
     $ gcloud compute instance-groups unmanaged add-instances $INFRA_ID-master-$ZONE_2-instance-group --zone=$ZONE_2 --instances=$INFRA_ID-master-2
- For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

### 4.9.15.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

**Example 4.8. 05_control_plane.py Deployment Manager template**

```python
def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-master-0',
         'type': 'compute.v1.instance',
         'properties': {
             'disks': [{
                 'autoDelete': True,
                 'boot': True,
                 'initializeParams': {
                     'diskSizeGb': context.properties['root_volume_size'],
                     'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                     'sourceImage': context.properties['image']
                 }
             }],
             'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' + context.properties['machine_type'],
             'metadata': {
                 'items': [{'key': 'user-data',
                             'value': context.properties['ignition']}
                 ],
             'networkInterfaces': [{
                 'subnetwork': context.properties['control_subnet']
             }],
             'serviceAccounts': [{
                 'email': context.properties['service_account_email'],
                 'scopes': ['https://www.googleapis.com/auth/cloud-platform']
             }],
             'tags': {
                 'items': [
                     context.properties['infra_id'] + '-master',
                 ],
             'zone': context.properties['zones'][0]
         }
    ]
```
'name': context.properties['infra_id'] + '-master-1',
'type': 'compute.v1.instance',
'properties': {
'disks': [[
  'autoDelete': True,
  'boot': True,
  'initializeParams': {
    'diskSizeGb': context.properties['root_volume_size'],
    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
    'sourceImage': context.properties['image']
  }
],
'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' + context.properties['machine_type'],
'metadata': {
  'items': [{
    'key': 'user-data',
    'value': context.properties['ignition']
  }]
},
'networkInterfaces': [{
  'subnetwork': context.properties['control_subnet']
}]
},
'serviceAccounts': [{
  'email': context.properties['service_account_email'],
  'scopes': ['https://www.googleapis.com/auth/cloud-platform']
}],
'tags': {
  'items': [
    context.properties['infra_id'] + '-master',
  ]
},
'zone': context.properties['zones'][1]
}
},
'name': context.properties['infra_id'] + '-master-2',
'type': 'compute.v1.instance',
'properties': {
'disks': [[
  'autoDelete': True,
  'boot': True,
  'initializeParams': {
    'diskSizeGb': context.properties['root_volume_size'],
    'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
    'sourceImage': context.properties['image']
  }
],
'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' + context.properties['machine_type'],
'metadata': {
  'items': [{
    'key': 'user-data',
    'value': context.properties['ignition']
  }]
},
'networkInterfaces': [{

4.9.16. Wait for bootstrap completion and remove bootstrap resources in GCP

After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \
   --log-level info
   ```

   - **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   - **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   If the command exits without a FATAL warning, your production control plane has initialized.
Delete the bootstrap resources:

```
$ gcloud compute backend-services remove-backend $(INFRA_ID)-api-internal-backend-service --region=${(REGION)} --instance-group=$(INFRA_ID)-bootstrap-instance-group --instance-group-zone=${ZONE_0}
$ gsutil rm gs://$(INFRA_ID)-bootstrap-ignition/bootstrap.ign
$ gsutil rb gs://$(INFIRA_ID)-bootstrap-ignition
$ gcloud deployment-manager deployments delete $(INFIRA_ID)-bootstrap
```

4.9.17. Creating additional worker machines in GCP

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type `06_worker.py` in the file.

NOTE

If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the Deployment Manager template for worker machines section of this topic and save it as `06_worker.py` on your computer. This template describes the worker machines that your cluster requires.

2. Export the variables that the resource definition uses.
   a. Export the subnet that hosts the compute machines:

   ```
   $ export COMPUTE_SUBNET=$(gcloud compute networks subnets describe $(INFRA_ID)-worker-subnet --region=${(REGION)} --format json | jq -r .selfLink`
   ```
b. Export the email address for your service account:

```bash
$ export WORKER_SERVICE_ACCOUNT="\$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}.\" | jq -r ".[0].email")"
```

c. Export the location of the compute machine Ignition config file:

```bash
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. Create a **06_worker.yaml** resource definition file:

```bash
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0'
  type: 06_worker.py
  properties:
    infra_id: "\${INFRA_ID}\"  # 2
    zone: "\${ZONE_0}\"  # 3
    compute_subnet: "\${COMPUTE_SUBNET}\"  # 4
    image: "\${CLUSTER_IMAGE}\"  # 5
    machine_type: 'n1-standard-4'  # 6
    root_volume_size: '128'
    service_account_email: "\${WORKER_SERVICE_ACCOUNT}\"  # 7
    ignition: "\${WORKER_IGNITION}\"  # 8

- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: "\${INFRA_ID}\"  # 9
    zone: "\${ZONE_1}\"  # 10
    compute_subnet: "\${COMPUTE_SUBNET}\"  # 11
    image: "\${CLUSTER_IMAGE}\"  # 12
    machine_type: 'n1-standard-4'  # 13
    root_volume_size: '128'
    service_account_email: "\${WORKER_SERVICE_ACCOUNT}\"  # 14
    ignition: "\${WORKER_IGNITION}\"  # 15
EOF
```

- **name** is the name of the worker machine, for example **worker-0**.
- **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.
- **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.
- **compute_subnet** is the selfLink URL to the compute subnet.
- **image** is the selfLink URL to the RHCOS image.
- **machine_type** is the machine type of the instance, for example **n1-standard-4**.
- **service_account_email** is the email address for the worker service account that you created.
8.15 \textbf{ignition} is the contents of the \texttt{worker.ign} file.

4. Optional: If you want to launch additional instances, include additional resources of type \texttt{06_worker.py} in your \texttt{06_worker.yaml} resource definition file.

5. Create the deployment by using the \texttt{gcloud} CLI:

\begin{verbatim}
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config 06_worker.yaml
\end{verbatim}

\subsection*{4.9.17.1. Deployment Manager template for worker machines}

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

\begin{quote}
\textbf{Example 4.9. 06_worker.py Deployment Manager template}

\begin{verbatim}
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' + context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
    }
\end{verbatim}
\end{quote}
4.9.18. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

4.9.18.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```bash
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```bash
$ oc <command>
```

4.9.18.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

   After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   C:\> oc <command>
   ```

### 4.9.18.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your **PATH**.
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the `oc` command:

   ```
   $ oc <command>
   ```

### 4.9.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the **kubeadmin** credentials:
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   $ oc whoami
   system:admin
   ```

4.9.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the Pending or Approved status for each machine that you added to the cluster:

   ```bash
   $ oc get csr
   ```

   **Example output**
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  
  ```
  $ oc adm certificate approve <csr_name>  
  ```

  1  
  
  <csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}"| xargs --no-run-if-empty oc adm certificate approve  
  ```

**NOTE**

Some Operators might not become available until some CSRs are approved.
4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:
  
  ```
  $ oc adm certificate approve <csr_name>
  ```

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).
4.9.21. Optional: Adding the ingress DNS records

If you removed the DNS zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the ingress load balancer. You can create either a wildcard *.apps.{baseDomain}, or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- Configure a GCP account.
- Remove the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

Procedure

1. Wait for the Ingress router to create a load balancer and populate the `EXTERNAL-IP` field:

   ```bash
   $ oc -n openshift-ingress get service router-default
   
   NAME             TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)                      AGE
   ```

2. Add the A record to your zones:

   - To use A records:
     
     i. Export the variable for the router IP address:

     ```bash
     $ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
     ```

     ii. Add the A record to the private zones:

     ```bash
     $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
     $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
     $ gcloud dns record-sets transaction add ${ROUTER_IP} --name ".apps.${CLUSTER_NAME}.${BASE_DOMAIN}.
     --ttl 300 --type A --zone ${INFRA_ID}-private-zone
     $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
     ```
iii. For an external cluster, also add the A record to the public zones:

```bash
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name \\*.apps.$[CLUSTER_NAME].${BASE_DOMAIN}. --ttl 300 --type A --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

- To add explicit domains instead of using a wildcard, create entries for each of the cluster’s current routes:

```bash
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host} {"n"}{end}{end}' routes
```

**Example output**

```
oauth-openshift.apps.your.cluster.domain.example.com
cache-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

### 4.9.22. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

**Prerequisites**

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.
- Install the `oc` CLI and log in.

**Procedure**

1. Complete the cluster installation:

```bash
./openshift-install --dir <installation_directory> wait-for install-complete
```

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
2. Observe the running state of your cluster.

   a. Run the following command to view the current cluster version and status:

   ```bash
   $ oc get clusterversion
   
   Example output
   
   NAME     VERSION   AVAILABLE   PROGRESSING   SINCE       STATUS
   version   False    True        24m         Working towards 4.5.4: 99% complete
   
   b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

   ```bash
   $ oc get clusteroperators
   
   Example output
   
   NAME                                 VERSION   AVAILABLE   PROGRESSING   DEGRADED SINCE
   authentication                       4.5.4     True        False         False      7m56s
   cloud-credential                    4.5.4     True        False         False      31m
   cluster-autoscaler                  4.5.4     True        False         False      16m
   console                             4.5.4     True        False         False      10m
   csi-snapshot-controller             4.5.4     True        False         False      16m
   dns                                  4.5.4     True        False         False      22m
   etcd                                 4.5.4     False       False         False      25s
   image-registry                       4.5.4     True        False         False      16m
   ingress                              4.5.4     True        False         False      16m
   insights                            4.5.4     True        False         False      17m
   kube-apiserver                       4.5.4     True        False         False      19m
   kube-controller-manager             4.5.4     True        False         False      20m
   kube-scheduler                       4.5.4     True        False         False      20m
   kube-storage-version-migrator       4.5.4     True        False         False      16m
   machine-api                          4.5.4     True        False         False      22m
   machine-config                       4.5.4     True        False         False      22m
   marketplace                          4.5.4     True        False         False      16m
   ```
Run the following command to view your cluster pods:

```
$ oc get pods --all-namespaces
```

### Example output

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-111.us-east-2</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>2.compute.internal</td>
<td>etcd-member-ip-10-0-3-239.us-east-2</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>37m</td>
</tr>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-24.us-east-2</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-6d6674f4f4-h7t2t</td>
<td>1/1</td>
<td>Running</td>
<td>1</td>
<td>37m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-fm48r</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-fxkvv</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>29m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-q85nm</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>29m</td>
</tr>
<tr>
<td>openshift-service-ca-operator</td>
<td>openshift-service-ca-operator-66ff6dc6cd-9r257</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>37m</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>apiservice-cabundle-injector-695b6bcbc-cl5hm</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>configmap-cabundle-injector-8498544d7-25qn6</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>service-serving-cert-signer-6445fc9c6-wqdqn</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>openshift-service-catalog-apiserver-operator</td>
<td>openshift-service-catalog-apiserver-operator-549f44668b-b5q2w</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>32m</td>
</tr>
<tr>
<td>openshift-service-catalog-controller-manager-operator</td>
<td>openshift-service-catalog-controller-manager-operator-b78cr2lnm</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>31m</td>
</tr>
</tbody>
</table>

When the current cluster version is **AVAILABLE**, the installation is complete.

### 4.9.23. Telemetry access for OpenShift Container Platform
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

4.9.24. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

4.10. INSTALLING A CLUSTER INTO A SHARED VPC ON GCP USING DEPLOYMENT MANAGER TEMPLATES

In OpenShift Container Platform version 4.6, you can install a cluster into a shared Virtual Private Cloud (VPC) on Google Cloud Platform (GCP) that uses infrastructure that you provide. In this context, a cluster installed into a shared VPC is a cluster that is configured to use a VPC from a project different from where the cluster is being deployed.

A shared VPC enables an organization to connect resources from multiple projects to a common VPC network. You can communicate within the organization securely and efficiently by using internal IPs from that network. For more information about shared VPC, see Shared VPC overview in the GCP documentation.

The steps for performing a user-provided infrastructure installation into a shared VPC are outlined here. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.

**IMPORTANT**

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

4.10.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.
- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.
NOTE

Be sure to also review this site list if you are configuring a proxy.

4.10.2. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The kube-controller-manager only approves the kubelet client CSRs. The machine-approver cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

4.10.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.10.4. Configuring the GCP project that hosts your cluster

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

4.10.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

Procedure

- Create a project to host your OpenShift Container Platform cluster. See Creating and Managing Projects in the GCP documentation.
IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the `api-int.<cluster_name>.<base_domain>` URL; the Premium Tier is required for internal load balancing.

4.10.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. See Enabling services in the GCP documentation.

Table 4.37. Required API services

<table>
<thead>
<tr>
<th>API service</th>
<th>Console service name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Deployment Manager V2 API</td>
<td>deploymentmanager.googleapis.com</td>
</tr>
<tr>
<td>Compute Engine API</td>
<td>compute.googleapis.com</td>
</tr>
<tr>
<td>Google Cloud APIs</td>
<td>cloudapis.googleapis.com</td>
</tr>
<tr>
<td>Cloud Resource Manager API</td>
<td>cloudresourcemanager.googleapis.com</td>
</tr>
<tr>
<td>Google DNS API</td>
<td>dns.googleapis.com</td>
</tr>
<tr>
<td>IAM Service Account Credentials API</td>
<td>iamcredentials.googleapis.com</td>
</tr>
<tr>
<td>Identity and Access Management (IAM) API</td>
<td>iam.googleapis.com</td>
</tr>
<tr>
<td>Service Management API</td>
<td>servicemanagement.googleapis.com</td>
</tr>
<tr>
<td>Service Usage API</td>
<td>serviceusage.googleapis.com</td>
</tr>
<tr>
<td>Google Cloud Storage JSON API</td>
<td>storage-api.googleapis.com</td>
</tr>
<tr>
<td>Cloud Storage</td>
<td>storage-component.googleapis.com</td>
</tr>
</tbody>
</table>

4.10.4.3. GCP account limits
The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default Quotas do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

**Table 4.38. GCP resources used in a default cluster**

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Location</th>
<th>Total resources required</th>
<th>Resources removed after bootstrap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service account</td>
<td>IAM</td>
<td>Global</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Firewall rules</td>
<td>Networking</td>
<td>Global</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>Forwarding rules</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Health checks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Images</td>
<td>Compute</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Networks</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routers</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routes</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Subnetworks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Target pools</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

**NOTE**

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
4.10.4.4. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See Creating a service account in the GCP documentation.

2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the Owner role to it. See Granting roles to a service account for specific resources.

   **NOTE**

   While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. Create the service account key in JSON format. See Creating service account keys in the GCP documentation.

   The service account key is required to create a cluster.

4.10.4.4.1. Required GCP permissions

When you attach the Owner role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. To deploy an OpenShift Container Platform cluster, the service account requires the following permissions. If you
deploy your cluster into an existing VPC, the service account does not require certain networking permissions, which are noted in the following lists:

**Required roles for the installation program**
- Compute Admin
- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin

**Required roles for creating network resources during installation**
- DNS Administrator

**Required roles for user-provisioned GCP infrastructure**
- Deployment Manager Editor
- Service Account Key Admin

**Optional roles**
For the cluster to create new limited credentials for its Operators, add the following role:
- Service Account Key Admin

The roles are applied to the service accounts that the control plane and compute machines use:

**Table 4.39. GCP service account permissions**

<table>
<thead>
<tr>
<th>Account</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Plane</td>
<td>roles/compute.instanceAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.networkAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.securityAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
<tr>
<td></td>
<td>roles/iam.serviceAccountUser</td>
</tr>
<tr>
<td>Compute</td>
<td>roles/compute.viewer</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
</tbody>
</table>

4.10.4.5. Supported GCP regions
You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- asia-east1 (Changhua County, Taiwan)
- asia-east2 (Hong Kong)
- asia-northeast1 (Tokyo, Japan)
- asia-northeast2 (Osaka, Japan)
- asia-northeast3 (Seoul, South Korea)
- asia-south1 (Mumbai, India)
- asia-southeast1 (Jurong West, Singapore)
- asia-southeast2 (Jakarta, Indonesia)
- australia-southeast1 (Sydney, Australia)
- europe-north1 (Hamina, Finland)
- europe-west1 (St. Ghislain, Belgium)
- europe-west2 (London, England, UK)
- europe-west3 (Frankfurt, Germany)
- europe-west4 (Eemshaven, Netherlands)
- europe-west6 (Zürich, Switzerland)
- northamerica-northeast1 (Montréal, Québec, Canada)
- southamerica-east1 (São Paulo, Brazil)
- us-central1 (Council Bluffs, Iowa, USA)
- us-east1 (Moncks Corner, South Carolina, USA)
- us-east4 (Ashburn, Northern Virginia, USA)
- us-west1 (The Dalles, Oregon, USA)
- us-west2 (Los Angeles, California, USA)
- us-west3 (Salt Lake City, Utah, USA)
- us-west4 (Las Vegas, Nevada, USA)

4.10.4.6. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

Prerequisites
You created a project to host your cluster.

You created a service account and granted it the required permissions.

Procedure

1. Install the following binaries in $PATH:
   - `gcloud`
   - `gsutil`

   See Install the latest Cloud SDK version in the GCP documentation.

2. Authenticate using the `gcloud` tool with your configured service account.
   See Authorizing with a service account in the GCP documentation.

4.10.5. Configuring the GCP project that hosts your shared VPC network

If you use a shared Virtual Private Cloud (VPC) to host your OpenShift Container Platform cluster in Google Cloud Platform (GCP), you must configure the project that hosts it.

**NOTE**

If you already have a project that hosts the shared VPC network, review this section to ensure that the project meets all of the requirements to install an OpenShift Container Platform cluster.

Procedure

1. Create a project to host the shared VPC for your OpenShift Container Platform cluster. See Creating and Managing Projects in the GCP documentation.

2. Create a service account in the project that hosts your shared VPC. See Creating a service account in the GCP documentation.

3. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the **Owner** role to it. See Granting roles to a service account for specific resources.
NOTE
While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

The service account for the project that hosts the shared VPC network requires the following roles:

- Compute Network User
- Compute Security Admin
- Deployment Manager Editor
- DNS Administrator
- Security Admin
- Network Management Admin

4.10.5.1. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the project that hosts the shared VPC that you install the cluster into. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.

   NOTE
   If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see Google Domains.

2. Create a public hosted zone for your domain or subdomain in your GCP project. See Creating public zones in the GCP documentation.
   Use an appropriate root domain, such as openshiftcorp.com, or subdomain, such as clusters.openshiftcorp.com.

3. Extract the new authoritative name servers from the hosted zone records. See Look up your Cloud DNS name servers in the GCP documentation.
   You typically have four name servers.

4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: How to switch to custom name servers.

5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See Migrating to Cloud DNS in the GCP documentation.
6. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain. This process might include a request to your company’s IT department or the division that controls the root domain and DNS services for your company.

## 4.10.5.2. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.

### NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

### Prerequisites

- Configure a GCP account.

### Procedure

1. Copy the template from the Deployment Manager template for the VPC section of this topic and save it as `01_vpc.py` on your computer. This template describes the VPC that your cluster requires.

2. Export the following variables required by the resource definition:
   a. Export the control plane CIDR:

   ```
   $ export MASTER_SUBNET_CIDR='10.0.0.0/19'
   ```

   b. Export the compute CIDR:

   ```
   $ export WORKER_SUBNET_CIDR='10.0.32.0/19'
   ```

   c. Export the region to deploy the VPC network and cluster to:

   ```
   $ export REGION='<region>'
   ```

3. Export the variable for the ID of the project that hosts the shared VPC:

   ```
   $ export HOST_PROJECT=<host_project>
   ```

4. Export the variable for the email of the service account that belongs to host project:

   ```
   $ export HOST_PROJECT_ACCOUNT=<host_service_account_email>
   ```

5. Create a `01_vpc.yaml` resource definition file:

   ```
   $ cat <<EOF >01_vpc.yaml
   imports:
   ```
infra_id is the prefix of the network name.

region is the region to deploy the cluster into, for example us-central1.

master_subnet_cidr is the CIDR for the master subnet, for example 10.0.0.0/19.

worker_subnet_cidr is the CIDR for the worker subnet, for example 10.0.32.0/19.

6. Create the deployment by using the gcloud CLI:

```bash
$ gcloud deployment-manager deployments create <vpc_deployment_name> --config 01_vpc.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

For `<vpc_deployment_name>`, specify the name of the VPC to deploy.

7. Export the VPC variable that other components require:

   a. Export the name of the host project network:

   ```bash
   $ export HOST_PROJECT_NETWORK=<vpc_network>
   ```

   b. Export the name of the host project control plane subnet:

   ```bash
   $ export HOST_PROJECT_CONTROL_SUBNET=<control_plane_subnet>
   ```

   c. Export the name of the host project compute subnet:

   ```bash
   $ export HOST_PROJECT_COMPUTE_SUBNET=<compute_subnet>
   ```

8. Set up the shared VPC. See Setting up Shared VPC in the GCP documentation.

4.10.5.2.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

```python
Example 4.10. 01_vpc.py Deployment Manager template
def GenerateConfig(context):
```

resources = [
    {'name': context.properties['infra_id'] + '-network',
     'type': 'compute.v1.network',
     'properties': {
         'region': context.properties['region'],
         'autoCreateSubnetworks': False
     }
    },
    {'name': context.properties['infra_id'] + '-master-subnet',
     'type': 'compute.v1.subnetwork',
     'properties': {
         'region': context.properties['region'],
         'network': '$(ref. + context.properties[infra_id] + -network.selfLink)',
         'ipCidrRange': context.properties['master_subnet_cidr']
     }
    },
    {'name': context.properties['infra_id'] + '-worker-subnet',
     'type': 'compute.v1.subnetwork',
     'properties': {
         'region': context.properties['region'],
         'network': '$(ref. + context.properties[infra_id] + -network.selfLink)',
         'ipCidrRange': context.properties['worker_subnet_cidr']
     }
    },
    {'name': context.properties['infra_id'] + '-router',
     'type': 'compute.v1.router',
     'properties': {
         'region': context.properties['region'],
         'network': '$(ref. + context.properties[infra_id] + -network.selfLink)',
         'nats': [
             {'name': context.properties['infra_id'] + '-nat-master',
              'natIpAllocateOption': 'AUTO_ONLY',
              'minPortsPerVm': 7168,
              'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
              'subnetworks': [{
                  'name': '$(ref. + context.properties[infra_id] + -master-subnet.selfLink)',
                  'sourceIpRangesToNat': ['ALL_IP_RANGES']
              }]}
         },
         {'name': context.properties['infra_id'] + '-nat-worker',
          'natIpAllocateOption': 'AUTO_ONLY',
          'minPortsPerVm': 512,
          'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
          'subnetworks': [{
              'name': '$(ref. + context.properties[infra_id] + -worker-subnet.selfLink)',
              'sourceIpRangesToNat': ['ALL_IP_RANGES']
          }]}
     }
    ]
]

return {'resources': resources}

4.10.6. Creating the installation files for GCP
To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the `install-config.yaml` file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate `var` partition during the preparation phases of installation.

### 4.10.6.1. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 4.10.6.2. Sample customized `install-config.yaml` file for GCP

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.
IMPORTANT

This sample YAML file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program and modify it.

```yaml
apiVersion: v1
baseDomain: example.com
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
compute:
  - hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 0
metadata:
  name: test-cluster
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
      networkType: OpenShiftSDN
      serviceNetwork:
      - 172.30.0.0/16
  platform:
    gcp:
      projectID: openshift-production
      region: us-central1
    pullSecret: '{"auths": ...}'
fips: false
sshKey: ssh-ed25519 AAA...
publish: Internal
```

1. Specify the public DNS on the host project.
2. If you do not provide these parameters and values, the installation program provides the default value.
3. The `controlPlane` section is a single mapping, but the `compute` section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen `-`, and the first line of the `controlPlane` section must not. Although both...
must begin with a hyphen, \textasciitilde{}, and the first line of the \texttt{control-plane} section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

4 Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to \texttt{Disabled}. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

\textbf{IMPORTANT}

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger machine types, such as \texttt{n1-standard-8}, for your machines if you disable simultaneous multithreading.

7 Specify the main project where the VM instances reside.

8 Specify the region that your VPC network is in.

9 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

\textbf{IMPORTANT}

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the \texttt{x86_64} architecture.

10 You can optionally provide the \texttt{sshKey} value that you use to access the machines in your cluster.

\textbf{NOTE}

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your \texttt{ssh-agent} process uses.

11 How to publish the user-facing endpoints of your cluster. Set \texttt{publish} to \texttt{Internal} to deploy a private cluster, which cannot be accessed from the Internet. The default value is \texttt{External}. To use a shared VPC in a cluster that uses infrastructure that you provision, you must set \texttt{publish} to \texttt{Internal}. The installation program will no longer be able to access the public DNS zone for the base domain in the host project.

4.10.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the \texttt{install-config.yaml} file.

\textbf{Prerequisites}
You have an existing `install-config.yaml` file.

You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> ①
     httpsProxy: https://<username>:<pswd>@<ip>:<port> ②
     noProxy: example.com  ③
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   ...
   ```

   ① A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

   ② A proxy URL to use for creating HTTPS connections outside the cluster.

   ③ A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use * to bypass the proxy for all destinations.

   ④ If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 4.10.6.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.

- You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   **Example output**
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift

1. For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
   ```

   By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the worker machines:

   ```bash
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the `mastersSchedulable` parameter in the
   `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

5. Remove the `privateZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yml` DNS configuration file:

   ```yaml
   apiVersion: config.openshift.io/v1
   kind: DNS
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     baseDomain: example.openshift.com
     privateZone: 1
     id: mycluster-100419-private-zone
     status: {}
   ```

   Remove this section completely.

6. Configure the cloud provider for your VPC.

   a. Open the `<installation_directory>/manifests/cloud-provider-config.yaml` file.

   b. Add the `network-project-id` parameter and set its value to the ID of project that hosts the shared VPC network.
c. Add the `network-name` parameter and set its value to the name of the shared VPC network that hosts the OpenShift Container Platform cluster.

d. Replace the value of the `subnetwork-name` parameter with the value of the shared VPC subnet that hosts your compute machines.

The contents of the `<installation_directory>/manifests/cloud-provider-config.yaml` resemble the following example:

```yaml
config: |
  [global]
  project-id = example-project
  regional = true
  multizone = true
  node-tags = opensh-ptzzx-master
  node-tags = opensh-ptzzx-worker
  node-instance-prefix = opensh-ptzzx
  external-instance-groups-prefix = opensh-ptzzx
  network-project-id = example-shared-vpc
  network-name = example-network
  subnetwork-name = example-worker-subnet
```

7. If you deploy a cluster that is not on a private network, open the `<installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml` file and replace the value of the `scope` parameter with `External`. The contents of the file resemble the following example:

```yaml
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
      type: LoadBalancerService
status:
  availableReplicas: 0
  domain: ""
  selector: ""
```

8. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```bash
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
```

---

897
4.10.7. Exporting common variables

4.10.7.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:
  $$
  
  $ jq -r .infraID <installation_directory>/metadata.json
  $$

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

  Example output

  openshift-vw9j6

  The output of this command is your cluster name and a random string.

4.10.7.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).

NOTE

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.
Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Generate the Ignition config files for your cluster.
- Install the `jq` package.

Procedure

1. Export the following common variables to be used by the provided Deployment Manager templates:

   ```bash
   $ export BASE_DOMAIN='<base_domain>'
   $ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
   $ export NETWORK_CIDR='10.0.0.0/16'
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   $ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
   $ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
   $ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
   
   1. Supply the values for the host project.
   2. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

4.10.8. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config from the machine config server.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 4.40. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports <strong>9100-9101</strong> and the Cluster Version Operator on port <strong>9099</strong>.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>Protocol</td>
<td>Port</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 4.41. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 4.42. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**
Do not configure session persistence for an API load balancer.
Configure the following ports on both the front and back of the load balancers:

### Table 4.43. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

### 2. Application Ingress load balancer

Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

### Table 4.44. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>
TIP
If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE
A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

4.10.9. Creating load balancers in GCP
You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

NOTE
If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites
- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure
1. Copy the template from the Deployment Manager template for the internal load balancer section of this topic and save it as 02_lb_int.py on your computer. This template describes the internal load balancing objects that your cluster requires.

2. For an external cluster, also copy the template from the Deployment Manager template for the external load balancer section of this topic and save it as 02_lb_ext.py on your computer. This template describes the external load balancing objects that your cluster requires.

3. Export the variables that the deployment template uses:
   a. Export the cluster network location:
   
   ```bash
   $ export CLUSTER_NETWORK=('gcloud compute networks describe
   ${HOST_PROJECT_NETWORK} --project ${HOST_PROJECT} --account
   ${HOST_PROJECT_ACCOUNT} --format json | jq -r .selfLink')
   ```

   b. Export the control plane subnet location:
   
   ```bash
   $ export CONTROL_SUBNET=('gcloud compute networks subnets describe
   ${HOST_PROJECT_CONTROL_SUBNET} --region=${REGION} --project
   ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
   .selfLink')
   ```
c. Export the three zones that the cluster uses:

```bash
$ export ZONE_0=$(gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9)

$ export ZONE_1=$(gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9)

$ export ZONE_2=$(gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9)
```

4. Create a `02_infra.yaml` resource definition file:

```yaml
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py

resources:
- name: cluster-lb-ext
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}'
    region: '${REGION}'

- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}'
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones:
      - '${ZONE_0}'
      - '${ZONE_1}'
      - '${ZONE_2}'

EOF
```

1. Required only when deploying an external cluster.

2. `infra_id` is the INFRA_ID infrastructure name from the extraction step.

3. `region` is the region to deploy the cluster into, for example `us-central1`.

4. `control_subnet` is the URI to the control subnet.

5. `zones` are the zones to deploy the control plane instances into, like `us-east1-b`, `us-east1-c`, and `us-east1-d`.

5. Create the deployment by using the `gcloud` CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:
For an external cluster, also export the cluster public IP address:

$ export CLUSTER_PUBLIC_IP="$(gcloud compute addresses describe $INFRA_ID-cluster-public-ip --region=${REGION} --format json | jq -r .address)"

7. For an external cluster, also export the cluster public IP address:

4.10.9.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

Example 4.11.02_lb_ext.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check probe for kube-apiserver
        {'name': context.properties['infra_id'] + '-api-http-health-check',
         'type': 'compute.v1.httpHealthCheck',
         'properties': {
             'port': 6080,
             'requestPath': '/readyz'
         }
    ],
    # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check probe for kube-apiserver
    {'name': context.properties['infra_id'] + '-api-target-pool',
     'type': 'compute.v1.targetPool',
     'properties': {
         'region': context.properties['region'],
         'healthChecks': ['$ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink'],
         'instances': []
     }
    ],
    {'name': context.properties['infra_id'] + '-api-forwarding-rule',
     'type': 'compute.v1.forwardingRule',
     'properties': {
         'region': context.properties['region'],
         'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
         'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
         'portRange': '6443'
     }
    ]
    return {'resources': resources}
```
4.10.9.2. Deployment Manager template for the internal load balancer

You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

**Example 4.12. 02_lb_int.py Deployment Manager template**

```python
def GenerateConfig(context):
    backends = []
    for zone in context.properties['zones']:
        backends.append(
            {'group': '${(ref. + context.properties[infra_id] + `-master-` + zone + `-instance-group` + 
            `.selfLink)'
            })
    resources = [
        {'name': context.properties['infra_id'] + `-cluster-ip`,
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    ],
    # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
    # probe for kube-apiserver
    {'name': context.properties['infra_id'] + `-api-internal-health-check`,
    'type': 'compute.v1.healthCheck',
    'properties': {
        'httpsHealthCheck': {
            'port': 6443,
            'requestPath': '/readyz'
        },
        'type': "HTTPS"
    }
    ],
    {'name': context.properties['infra_id'] + `-api-internal-backend-service`,
    'type': 'compute.v1.regionBackendService',
    'properties': {
        'backends': backends,
        'healthChecks': ['$ref. + context.properties[infra_id] + `-api-internal-health-check.selfLink`],
        'loadBalancingScheme': 'INTERNAL',
        'region': context.properties['region'],
        'protocol': 'TCP',
        'timeoutSec': 120
    }
    ],
    {'name': context.properties['infra_id'] + `-api-internal-forwarding-rule`,
    'type': 'compute.v1.forwardingRule',
    'properties': {
        'backendService': '${(ref. + context.properties[infra_id] + `-api-internal-backend-service.selfLink`)',
        'IPAddress': '${(ref. + context.properties[infra_id] + `-cluster-ip.selfLink`),
        'loadBalancingScheme': 'INTERNAL',
```

You will need this template in addition to the `02_lb_ext.py` template when you create an external cluster.

### 4.10.10. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

```python
'ports': [6443, 22623],
'region': context.properties['region'],
'subnetwork': context.properties['control_subnet']
]
]]

for zone in context.properties['zones']:
    resources.append({
        'name': context.properties['infra_id'] + '-master-' + zone + '-instance-group',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namePorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                },
                {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': zone
        }
    })

return {'resources': resources}
```
1. Copy the template from the Deployment Manager template for the private DNS section of this topic and save it as `02_dns.py` on your computer. This template describes the private DNS objects that your cluster requires.

2. Create a `02_dns.yaml` resource definition file:

   ```yaml
   $ cat <<EOF >02_dns.yaml
   imports:
   - path: 02_dns.py

   resources:
   - name: cluster-dns
     type: 02_dns.py
     properties:
     infra_id: '${INFRA_ID}'
     cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}'
     cluster_network: '${CLUSTER_NETWORK}'
   EOF
   ```

   1. `infra_id` is the INFRA_ID infrastructure name from the extraction step.
   2. `cluster_domain` is the domain for the cluster, for example `openshift.example.com`.
   3. `cluster_network` is the selfLink URL to the cluster network.

3. Create the deployment by using the `gcloud` CLI:

   ```bash
   $ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   ```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

   a. Add the internal DNS entries:

   ```bash
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   ```

   b. For an external cluster, also add the external DNS entries:

   ```bash
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
   ```
4.10.10.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

Example 4.13. 02_dns.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-private-zone',
         'type': 'dns.v1.managedZone',
         'properties': {
             'description': '',
             'dnsName': context.properties['cluster_domain'] + '.',
             'visibility': 'private',
             'privateVisibilityConfig': {
                 'networks': [
                     {'networkUrl': context.properties['cluster_network']}
                 ]
             }
         }
    ]
    return {'resources': resources}
```

4.10.11. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
Procedure

1. Copy the template from the Deployment Manager template for firewall rules section of this topic and save it as 03_firewall.py on your computer. This template describes the security groups that your cluster requires.

2. Create a 03_firewall.yaml resource definition file:

   ```
   $ cat <<EOF >03_firewall.yaml
   imports:
   - path: 03_firewall.py
   
   resources:
   - name: cluster-firewall
     type: 03_firewall.py
     properties:
       allowed_external_cidr: '0.0.0.0/0'  
       infra_id: '${INFRA_ID}'
       cluster_network: '${CLUSTER_NETWORK}'
       network_cidr: '${NETWORK_CIDR}'
   EOF
   ```

   - **allowed_external_cidr** is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to ${NETWORK_CIDR}.
   - **infra_id** is the INFRA_ID infrastructure name from the extraction step.
   - **cluster_network** is the selfLink URL to the cluster network.
   - **network_cidr** is the CIDR of the VPC network, for example 10.0.0.0/16.

3. Create the deployment by using the gcloud CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config 03_firewall.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
   ```

4.10.11.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

Example 4.14. 03_firewall.py Deployment Manager template

```py
def GenerateConfig(context):

    resources = [
        {
            'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
            'type': 'compute.v1.firewall',
            'properties': {
                'network': context.properties['cluster_network'],
                'allowed': [{
                    'IPProtocol': 'tcp',
                    'ports': ['22']
                }],
            },
        }
    ]
```
'sourceRanges': [context.properties["allowed_external_cidr"]],
'targetTags': [context.properties["infra_id"] + '-bootstrap']
}
},

{name}: context.properties["infra_id"] + '-api',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties["cluster_network"],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['6443']
  }],
  'sourceRanges': [context.properties["allowed_external_cidr"]],
  'targetTags': [context.properties["infra_id"] + '-master']
}
},

{name}: context.properties["infra_id"] + '-health-checks',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties["cluster_network"],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['6080', '6443', '22624']
  }],
  'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
  'targetTags': [context.properties["infra_id"] + '-master']
}
},

{name}: context.properties["infra_id"] + '-etcd',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties["cluster_network"],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['2379-2380']
  }],
  'sourceTags': [context.properties["infra_id"] + '-master'],
  'targetTags': [context.properties["infra_id"] + '-master']
}
},

{name}: context.properties["infra_id"] + '-control-plane',
'type': 'compute.v1.firewall',
'properties': {
  'network': context.properties["cluster_network"],
  'allowed': [{
    'IPProtocol': 'tcp',
    'ports': ['10257']
  }],
  'sourceTags': [context.properties["infra_id"] + '-master'],
  'targetTags': [context.properties["infra_id"] + '-master']
}
CHAPTER 4. INSTALLING ON GCP

```yaml
context.properties['infra_id'] + '-worker'
},
'targetTags': [context.properties['infra_id'] + '-master']
}
{
'name': context.properties['infra_id'] + '-internal-network',
'type': 'compute.v1.firewall',
'properties': {
'network': context.properties['cluster_network'],
'allowed': [{
'IPProtocol': 'icmp'
},
{'IPProtocol': 'tcp',
'ports': ['22']
}],
'sourceRanges': [context.properties['network_cidr']],
'targetTags': [
context.properties['infra_id'] + '-master',
context.properties['infra_id'] + '-worker'
]
}
},
{
'name': context.properties['infra_id'] + '-internal-cluster',
'type': 'compute.v1.firewall',
'properties': {
'network': context.properties['cluster_network'],
'allowed': [{
'IPProtocol': 'udp',
'ports': ['4789', '6081']
},
{'IPProtocol': 'tcp',
'ports': ['9000-9999']
},
{'IPProtocol': 'udp',
'ports': ['9000-9999']
},
{'IPProtocol': 'tcp',
'ports': ['10250']
},
{'IPProtocol': 'tcp',
'ports': ['30000-32767']
},
{'IPProtocol': 'udp',
'ports': ['30000-32767']
}],
'sourceTags': [
context.properties['infra_id'] + '-master',
context.properties['infra_id'] + '-worker'
]
},
targetTags': [
context.properties['infra_id'] + '-master',
context.properties['infra_id'] + '-worker'
]
```
4.10.12. Creating IAM roles in GCP

You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

1. Copy the template from the Deployment Manager template for IAM roles section of this topic and save it as **03_iam.py** on your computer. This template describes the IAM roles that your cluster requires.

2. Create a **03_iam.yaml** resource definition file:

   ```
   imports:
   - path: 03_iam.py
   resources:
   - name: cluster-iam
     type: 03_iam.py
     properties:
       infra_id: '${INFRA_ID}'
   ```

3. Create the deployment by using the **gcloud** CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
   ```

4. Export the variable for the master service account:

   ```
   $ export GOOGLE_APPLICATION_CREDENTIALS=.../service-account.json
   ```
5. Export the variable for the worker service account:

   ```bash
   $ export WORKER_SERVICE_ACCOUNT=('gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r ".[0].email")
   ```

6. Assign the permissions that the installation program requires to the service accounts for the subnets that host the control plane and compute subnets:

   a. Grant the **networkViewer** role of the project that hosts your shared VPC to the master service account:

   ```bash
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} projects add-iam-policy-binding ${HOST_PROJECT} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkViewer"
   ```

   b. Grant the **networkUser** role to the master service account for the control plane subnet:

   ```bash
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} compute networks subnets add-iam-policy-binding "${HOST_PROJECT_CONTROL_SUBNET}" --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser" --region ${REGION}
   ```

   c. Grant the **networkUser** role to the worker service account for the control plane subnet:

   ```bash
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} compute networks subnets add-iam-policy-binding "${HOST_PROJECT_CONTROL_SUBNET}" --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser" --region ${REGION}
   ```

   d. Grant the **networkUser** role to the master service account for the compute subnet:

   ```bash
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} compute networks subnets add-iam-policy-binding "${HOST_PROJECT_COMPUTE_SUBNET}" --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser" --region ${REGION}
   ```

   e. Grant the **networkUser** role to the worker service account for the compute subnet:

   ```bash
   $ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} compute networks subnets add-iam-policy-binding "${HOST_PROJECT_COMPUTE_SUBNET}" --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser" --region ${REGION}
   ```

7. The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding $(PROJECT_NAME) --member "serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

8. Create a service account key and store it locally for later use:

$ gcloud iam service-accounts keys create service-account-key.json --iam-account=${MASTER_SERVICE_ACCOUNT}

4.10.12.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

Example 4.15. 03_iam.py Deployment Manager template

```python
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

4.10.13. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.
Procedure

1. Obtain the RHCOS image from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

   The file name contains the OpenShift Container Platform version number in the format `rhcos-<version>-<arch>-gcp.<arch>.tar.gz`.

2. Create the Google storage bucket:

   ```
   $ gsutil mb gs://<bucket_name>
   ```

3. Upload the RHCOS image to the Google storage bucket:

   ```
   $ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz gs://<bucket_name>
   ```

4. Export the uploaded RHCOS image location as a variable:

   ```
   $ export IMAGE_SOURCE=`gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz`
   ```

5. Create the cluster image:

   ```
   $ gcloud compute images create "${INFRA_ID}-rhcos-image" \
   --source-uri="${IMAGE_SOURCE}"
   ```

### 4.10.14. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
Create and configure a VPC and associated subnets in GCP.

Create and configure networking and load balancers in GCP.

Create control plane and compute roles.

Ensure pyOpenSSL is installed.

**Procedure**

1. Copy the template from the Deployment Manager template for the bootstrap machine section of this topic and save it as `04_bootstrap.py` on your computer. This template describes the bootstrap machine that your cluster requires.

2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:
   
   ```bash
   $ export CLUSTER_IMAGE=('gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink')
   ```

3. Create a bucket and upload the `bootstrap.ign` file:
   
   ```bash
   $ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
   $ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
   ```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:
   
   ```bash
   $ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
   ```

5. Create a `04_bootstrap.yaml` resource definition file:

   ```bash
   $ cat <<EOF >04_bootstrap.yaml
   imports:
   - path: 04_bootstrap.py
   
   resources:
   - name: cluster-bootstrap
     type: 04_bootstrap.py
     properties:
       infra_id: '${INFRA_ID}'
       region: '${REGION}'
       zone: '${ZONE_0}'
       cluster_network: '${CLUSTER_NETWORK}'
       control_subnet: '${CONTROL_SUBNET}'
       image: '${CLUSTER_IMAGE}'
       machine_type: 'n1-standard-4'
       root_volume_size: '128'
       bootstrap_ign: '${BOOTSTRAP_IGN}'
   EOF
   ```
**infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

**region** is the region to deploy the cluster into, for example **us-central1**.

**zone** is the zone to deploy the bootstrap instance into, for example **us-central1-b**.

**cluster_network** is the **selfLink** URL to the cluster network.

**control_subnet** is the **selfLink** URL to the control subnet.

**image** is the **selfLink** URL to the RHCOS image.

**machine_type** is the machine type of the instance, for example **n1-standard-4**.

**root_volume_size** is the boot disk size for the bootstrap machine.

**bootstrap_ign** is the URL output when creating a signed URL.

6. Create the deployment by using the **gcloud** CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config 04_bootstrap.yaml
```

7. Add the bootstrap instance to the internal load balancer instance group:

```bash
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-bootstrap-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

8. Add the bootstrap instance group to the internal load balancer backend service:

```bash
$ gcloud compute backend-services add-backend ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
```

### 4.10.14.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

#### Example 4.16. 04_bootstrap.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {
            'name': context.properties['infra_id'] + '-bootstrap-public-ip',
            'type': 'compute.v1.address',
            'properties': {
                'region': context.properties['region']
            }
        },
        {
            'name': context.properties['infra_id'] + '-bootstrap',
            'type': 'compute.v1.instance',
            'properties': {
                'disks': [
            ```
4.10.15. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.

```yaml
return {
  'resources': resources}
```
Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the Deployment Manager template for control plane machines section of this topic and save it as `05_control_plane.py` on your computer. This template describes the control plane machines that your cluster requires.

2. Export the following variable required by the resource definition:

   ```bash
   $ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
   ``

3. Create a `05_control_plane.yaml` resource definition file:

   ```bash
   $ cat <<<EOF >05_control_plane.yaml
   imports:
   - path: 05_control_plane.py
   resources:
   - name: cluster-control-plane
     type: 05_control_plane.py
     properties:
       infrad: '{{INFRA_ID}}'
       zones:
       - '${ZONE_0}'
       - '${ZONE_1}'
       - '${ZONE_2}'
       control_subnet: '${CONTROL_SUBNET}'
       image: '${CLUSTER_IMAGE}'
       machine_type: 'n1-standard-4'
       root_volume_size: '128'
       service_account_email: '${MASTER_SERVICE_ACCOUNT}'
       ignition: '${MASTER_IGNITION}'
   EOF
   ```
infra_id is the INFRA_ID infrastructure name from the extraction step.

zones are the zones to deploy the control plane instances into, for example us-central1-a, us-central1-b, and us-central1-c.

control_subnet is the selfLink URL to the control subnet.

image is the selfLink URL to the RHCOS image.

machine_type is the machine type of the instance, for example n1-standard-4.

service_account_email is the email address for the master service account that you created.

ignition is the contents of the master.ign file.

4. Create the deployment by using the gcloud CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
  05_control_plane.yaml
```

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

- Run the following commands to add the control plane machines to the appropriate instance groups:

```bash
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
  ${{ZONE_0}}-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
  ${{ZONE_1}}-instance-group --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
  ${{ZONE_2}}-instance-group --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

- For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

```bash
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
  zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
  zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
  zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

4.10.15.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

```python
Example 4.17. 05_control_plane.py Deployment Manager template

def GenerateConfig(context):
```
resources = [
    {
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [
                {'autoDelete': True, 'boot': True, 'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }],
                'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' + context.properties['machine_type'],
                'metadata': {
                    'items': [{'key': 'user-data', 'value': context.properties['ignition']}]
                },
                'networkInterfaces': [{'subnetwork': context.properties['control_subnet']}],
                'serviceAccounts': [{'email': context.properties['service_account_email'], 'scopes': ['https://www.googleapis.com/auth/cloud-platform']}
            ],
            'tags': [
                context.properties['infra_id'] + '-master',
            ],
            'zone': context.properties['zones'][0]
        }
    },
    {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [
                {'autoDelete': True, 'boot': True, 'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }],
                'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' + context.properties['machine_type'],
                'metadata': {
                    'items': [{'key': 'user-data', 'value': context.properties['ignition']}]
                }
            ]
        }
    }
]
4.10.16. Wait for bootstrap completion and remove bootstrap resources in GCP
After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

**Procedure**

1. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
   --log-level info 2
   ``

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   If the command exits without a FATAL warning, your production control plane has initialized.

2. Delete the bootstrap resources:

   ```bash
   $ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
   $ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
   $ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
   $ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
   ``

**4.10.17. Creating additional worker machines in GCP**

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type `06_worker.py` in the file.
NOTE
If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites
- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure
1. Copy the template from the Deployment Manager template for worker machines section of this topic and save it as 06_worker.py on your computer. This template describes the worker machines that your cluster requires.

2. Export the variables that the resource definition uses.
   a. Export the subnet that hosts the compute machines:

      ```
      $ export COMPUTE_SUBNET=`gcloud compute networks subnets describe
      ${HOST_PROJECT_COMPUTE_SUBNET} --region=${REGION} --project
      ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
      .selfLink`
      ```

   b. Export the email address for your service account:

      ```
      $ export WORKER_SERVICE_ACCOUNT=`gcloud iam service-accounts list --filter
      "email~'^${INFRA_ID}-w@${PROJECT_NAME}.'" --format json | jq -r '[0].email'`
      ```

   c. Export the location of the compute machine Ignition config file:

      ```
      $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
      ```

3. Create a 06_worker.yaml resource definition file:

      ```
      $ cat <<EOF >06_worker.yaml
      imports:
      - path: 06_worker.py

      resources:
      - name: 'worker-0'
      ```
type: 06_worker.py
properties:
  infra_id: '${INFRA_ID}'
  zone: '${ZONE_0}'
  compute_subnet: '${COMPUTE_SUBNET}'
  image: '${CLUSTER_IMAGE}'
  machine_type: 'n1-standard-4'
  root_volume_size: '128'
  service_account_email: '${WORKER_SERVICE_ACCOUNT}'
  ignition: '${WORKER_IGNITION}'
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}'
    zone: '${ZONE_1}'
    compute_subnet: '${COMPUTE_SUBNET}'
    image: '${CLUSTER_IMAGE}'
    machine_type: 'n1-standard-4'
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}'
    ignition: '${WORKER_IGNITION}'
EOF

1. **name** is the name of the worker machine, for example **worker-0**.

2. **infra_id** is the INFRA_ID infrastructure name from the extraction step.

3. **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.

4. **compute_subnet** is the selfLink URL to the compute subnet.

5. **image** is the selfLink URL to the RHCOS image.

6. **machine_type** is the machine type of the instance, for example **n1-standard-4**.

7. **service_account_email** is the email address for the worker service account that you created.

8. **ignition** is the contents of the **worker.ign** file.

4. Optional: If you want to launch additional instances, include additional resources of type **06_worker.py** in your **06_worker.yaml** resource definition file.

5. Create the deployment by using the **gcloud** CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config 06_worker.yaml
   ```

**4.10.17.1. Deployment Manager template for worker machines**

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:
Example 4.18. 06_worker.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                        'autoDelete': True,
                        'boot': True,
                        'initializeParams': {
                            'diskSizeGb': context.properties['root_volume_size'],
                            'sourceImage': context.properties['image']
                        }
                    },
                    'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' + context.properties['machine_type'],
                    'metadata': {
                        'items': [
                            {'key': 'user-data',
                             'value': context.properties['ignition']
                            }
                        ],
                        'networkInterfaces': [{
                            'subnetwork': context.properties['compute_subnet']
                        }]
                    },
                    'serviceAccounts': [{
                        'email': context.properties['service_account_email'],
                        'scopes': ['https://www.googleapis.com/auth/cloud-platform']
                    }],
                    'tags': {
                        'items': [
                            context.properties['infra_id'] + '-worker',
                        ],
                        'zone': context.properties['zone']
                    }
                }
            ],
            'zone': context.properties['zone']
        }
    ]
    return {'resources': resources}
```

4.10.18. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

4.10.18.1. Installing the OpenShift CLI on Linux
You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:

   $ tar xzvf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

4.10.18.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:

C:\> oc <command>

4.10.18.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.
Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

4.10.19. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

4.10.20. Approving the certificate signing requests for your machines
When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  63m  v1.19.0
   master-1  Ready     master  63m  v1.19.0
   master-2  Ready     master  64m  v1.19.0
   ```

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```bash
   $ oc get csr
   ```

   **Example output**

   ```
   NAME        AGE     REQUESTOR                                                                 CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   ...  
   ```

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:
NOTE
Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the machine-approver if the Kubelet requests a new certificate with identical parameters.

NOTE
For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the oc exec, oc rsh, and oc logs commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the system:node or system:admin groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  $ oc adm certificate approve <csr_name>

  <csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" xargs --no-run-if-empty oc adm certificate approve

NOTE
Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

  $ oc get csr

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
CHAPTER 4. INSTALLING ON GCP

5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for
your cluster machines:
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

To approve all pending CSRs, run the following command:
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
6. After all client and server CSRs have been approved, the machines have the Ready status.
Verify this by running the following command:
$ oc get nodes

Example output
NAME
master-0
master-1
master-2
worker-0
worker-1

STATUS ROLES AGE VERSION
Ready master 73m v1.20.0
Ready master 73m v1.20.0
Ready master 74m v1.20.0
Ready worker 11m v1.20.0
Ready worker 11m v1.20.0

NOTE
It can take a few minutes after approval of the server CSRs for the machines to
transition to the Ready status.
Additional information
For more information on CSRs, see Certificate Signing Requests .

4.10.21. Adding the ingress DNS records
DNS zone configuration is removed when creating Kubernetes manifests and generating Ignition
configs. You must manually create DNS records that point at the ingress load balancer. You can create
either a wildcard *.apps.{baseDomain}. or specific records. You can use A, CNAME, and other records
per your requirements.
Prerequisites
Configure a GCP account.
Remove the DNS Zone configuration when creating Kubernetes manifests and generating
Ignition configs.
Create and configure a VPC and associated subnets in GCP.

931


- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

**Procedure**

1. Wait for the Ingress router to create a load balancer and populate the `EXTERNAL-IP` field:

   ```bash
   $ oc -n openshift-ingress get service router-default
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
</table>

2. Add the A record to your zones:
   - To use A records:
     
     i. Export the variable for the router IP address:

     ```bash
     $ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
     ```

     ii. Add the A record to the private zones:

     ```bash
     $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     $ gcloud dns record-sets transaction add ${ROUTER_IP} --name ".apps.${CLUSTER_NAME}.${BASE_DOMAIN}.*" --ttl 300 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     ```

     iii. For an external cluster, also add the A record to the public zones:

     ```bash
     $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
     $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     $ gcloud dns record-sets transaction add ${ROUTER_IP} --name ".apps.${CLUSTER_NAME}.${BASE_DOMAIN}.*" --ttl 300 --type A --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
     ```
To add explicit domains instead of using a wildcard, create entries for each of the cluster’s current routes:

```bash
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{\"\n\"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

4.10.22. Adding ingress firewall rules

The cluster requires several firewall rules. If you do not use a shared VPC, these rules are created by the ingress controller via the GCP cloud provider. When you use a shared VPC, you can either create cluster-wide firewall rules for all services now or create each rule based on events, when the cluster requests access. By creating each rule when the cluster requests access, you know exactly which firewall rules are required. By creating cluster-wide firewall rules, you can apply the same rule set across multiple clusters.

If you choose to create each rule based on events, you must create firewall rules after you provision the cluster and during the life of the cluster when the console notifies you that rules are missing. Events that are similar to the following event are displayed, and you must add the firewall rules that are required:

```bash
$ oc get events -n openshift-ingress --field-selector="reason=LoadBalancerManualChange"
```

Example output

```
Firewall change required by security admin: `gcloud compute firewall-rules create k8s-fw-a26e631036a3f46cbaf8df67266d55 --network example-network --description "{"kubernetes.io/service-name":"openshift-ingress/router-default", "kubernetes.io/service-ip":\"35.237.236.234\"}" --allow tcp:443,tcp:80 --source-ranges 0.0.0.0/0 --target-tags exampl-fqzq7-master,exampl-fqzq7-worker --project example-project`
```

If you encounter issues when creating these rule-based events, you can configure the cluster-wide firewall rules while your cluster is running.

4.10.22.1. Creating cluster-wide firewall rules for a shared VPC in GCP

You can create cluster-wide firewall rules to allow the access that the OpenShift Container Platform cluster requires.
Prerequisites

- You exported the variables that the Deployment Manager templates require to deploy your cluster.
- You created the networking and load balancing components in GCP that your cluster requires.

Procedure

1. Add a single firewall rule to allow the Google Cloud Engine health checks to access all of the services. This rule enables the ingress load balancers to determine the health status of their instances.

   ```
   $ gcloud compute firewall-rules create --allow='tcp:30000-32767,udp:30000-32767' --network="${CLUSTER_NETWORK}" --source-ranges='130.211.0.0/22,35.191.0.0/16,209.85.152.0/22,209.85.204.0/22' --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress-hc --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
   ```

2. Add a single firewall rule to allow access to all cluster services:

   - For an external cluster:

     ```
     $ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --network="${CLUSTER_NETWORK}" --source-ranges="0.0.0.0/0" --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
     ```

   - For a private cluster:

     ```
     $ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --network="${CLUSTER_NETWORK}" --source-ranges=${NETWORK_CIDR} --target-tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
     ```

   Because this rule only allows traffic on TCP ports 80 and 443, ensure that you add all the ports that your services use.

4.10.23. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites
Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.

Install the `oc` CLI and log in.

**Procedure**

1. Complete the cluster installation:

   ```
   $ ./openshift-install --dir <installation_directory> wait-for install-complete
   ```

   **Example output**

   ```
   INFO Waiting up to 30m0s for the cluster to initialize...
   ```

   **For `<installation_directory>`**, specify the path to the directory that you stored the installation files in.

   **IMPORTANT**

   - The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

   - It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Observe the running state of your cluster.

   a. Run the following command to view the current cluster version and status:

   ```
   $ oc get clusterversion
   ```

   **Example output**

   ```
   NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
   version             False       True          24m     Working towards 4.5.4: 99% complete
   ```

   b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

   ```
   $ oc get clusteroperators
   ```

   **Example output**
Run the following command to view your cluster pods:

```bash
$ oc get pods --all-namespaces
```

### Example output

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-111.us-east-2.compute.internal</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-239.us-east-2.compute.internal</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>37m</td>
</tr>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-24.us-east-2.compute.internal</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
</tr>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-6d6674f4f4-h712t</td>
<td>1/1</td>
<td>Running</td>
<td>1</td>
<td>37m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiser-ver-fm48r</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiser-ver-fxkvv</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>29m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiser-ver-q85nm</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>29m</td>
</tr>
</tbody>
</table>
4.10.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources
- See About remote health monitoring for more information about the Telemetry service

4.10.25. Next steps
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

4.11. INSTALLING A CLUSTER ON GCP IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on Google Cloud Platform (GCP) that uses infrastructure that you provide and an internal mirror of the installation release content.

IMPORTANT

While you can install an OpenShift Container Platform cluster by using mirrored installation release content, your cluster still requires internet access to use the GCP APIs.

The steps for performing a user-provided infrastructure install are outlined here. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.
The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several Deployment Manager templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

4.11.1. Prerequisites

- Create a registry on your mirror host and obtain the `imageContentSources` data for your version of OpenShift Container Platform.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to. While you might need to grant access to more sites, you must grant access to `*.googleapis.com` and `accounts.google.com`.

- If you do not allow the system to manage identity and access management (IAM), then a cluster administrator can manually create and maintain IAM credentials. Manual mode can also be used in environments where the cloud IAM APIs are not reachable.

4.11.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

- **IMPORTANT**
  
  Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.
4.11.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

4.11.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access **Quay.io** to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

4.11.4. Configuring your GCP project

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

4.11.4.1. Creating a GCP project

To install OpenShift Container Platform, you must create a project in your Google Cloud Platform (GCP) account to host the cluster.

**Procedure**

- Create a project to host your OpenShift Container Platform cluster. See **Creating and Managing Projects** in the GCP documentation.
IMPORTANT

Your GCP project must use the Premium Network Service Tier if you are using installer-provisioned infrastructure. The Standard Network Service Tier is not supported for clusters installed using the installation program. The installation program configures internal load balancing for the `api-int.<cluster_name>.<base_domain>` URL; the Premium Tier is required for internal load balancing.

4.11.4.2. Enabling API services in GCP

Your Google Cloud Platform (GCP) project requires access to several API services to complete OpenShift Container Platform installation.

Prerequisites

- You created a project to host your cluster.

Procedure

- Enable the following required API services in the project that hosts your cluster. See Enabling services in the GCP documentation.

<table>
<thead>
<tr>
<th>API service</th>
<th>Console service name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute Engine API</td>
<td><code>compute.googleapis.com</code></td>
</tr>
<tr>
<td>Google Cloud APIs</td>
<td><code>cloudapis.googleapis.com</code></td>
</tr>
<tr>
<td>Cloud Resource Manager API</td>
<td><code>cloudresourcemanager.googleapis.com</code></td>
</tr>
<tr>
<td>Google DNS API</td>
<td><code>dns.googleapis.com</code></td>
</tr>
<tr>
<td>IAM Service Account Credentials API</td>
<td><code>iamcredentials.googleapis.com</code></td>
</tr>
<tr>
<td>Identity and Access Management (IAM) API</td>
<td><code>iam.googleapis.com</code></td>
</tr>
<tr>
<td>Service Management API</td>
<td><code>servicemanagement.googleapis.com</code></td>
</tr>
<tr>
<td>Service Usage API</td>
<td><code>serviceusage.googleapis.com</code></td>
</tr>
<tr>
<td>Google Cloud Storage JSON API</td>
<td><code>storage-api.googleapis.com</code></td>
</tr>
<tr>
<td>Cloud Storage</td>
<td><code>storage-component.googleapis.com</code></td>
</tr>
</tbody>
</table>

4.11.4.3. Configuring DNS for GCP

To install OpenShift Container Platform, the Google Cloud Platform (GCP) account you use must have a dedicated public hosted zone in the same project that you host the OpenShift Container Platform in.
cluster. This zone must be authoritative for the domain. The DNS service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through GCP or another source.

   **NOTE**
   
   If you purchase a new domain, it can take time for the relevant DNS changes to propagate. For more information about purchasing domains through Google, see [Google Domains](https://domains.google.com/).

2. Create a public hosted zone for your domain or subdomain in your GCP project. See [Creating public zones](https://cloud.google.com/dns/docs/public-zones) in the GCP documentation.

   Use an appropriate root domain, such as `openshiftcorp.com`, or subdomain, such as `clusters.openshiftcorp.com`.

3. Extract the new authoritative name servers from the hosted zone records. See [Look up your Cloud DNS name servers](https://cloud.google.com/dns/docs/using-name-servers) in the GCP documentation.

   You typically have four name servers.

4. Update the registrar records for the name servers that your domain uses. For example, if you registered your domain to Google Domains, see the following topic in the Google Domains Help: [How to switch to custom name servers](https://support.google.com/domains/answer/6134815).

5. If you migrated your root domain to Google Cloud DNS, migrate your DNS records. See [Migrating to Cloud DNS](https://cloud.google.com/dns/docs/migrating-dns) in the GCP documentation.

6. If you use a subdomain, follow your company’s procedures to add its delegation records to the parent domain. This process might include a request to your company’s IT department or the division that controls the root domain and DNS services for your company.

4.11.4.4. GCP account limits

The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default Quotas do not affect your ability to install a default OpenShift Container Platform cluster.

A default cluster, which contains three compute and three control plane machines, uses the following resources. Note that some resources are required only during the bootstrap process and are removed after the cluster deploys.

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Location</th>
<th>Total resources required</th>
<th>Resources removed after bootstrap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service account</td>
<td>IAM</td>
<td>Global</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Firewall rules</td>
<td>Networking</td>
<td>Global</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>Service</td>
<td>Component</td>
<td>Location</td>
<td>Total resources required</td>
<td>Resources removed after bootstrap</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>----------</td>
<td>--------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Forwarding rules</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Health checks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Images</td>
<td>Compute</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Networks</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routers</td>
<td>Networking</td>
<td>Global</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Routes</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Subnetworks</td>
<td>Compute</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Target pools</td>
<td>Networking</td>
<td>Global</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

**NOTE**

If any of the quotas are insufficient during installation, the installation program displays an error that states both which quota was exceeded and the region.

Be sure to consider your actual cluster size, planned cluster growth, and any usage from other clusters that are associated with your account. The CPU, static IP addresses, and persistent disk SSD (storage) quotas are the ones that are most likely to be insufficient.

If you plan to deploy your cluster in one of the following regions, you will exceed the maximum storage quota and are likely to exceed the CPU quota limit:

- `asia-east2`
- `asia-northeast2`
- `asia-south1`
- `australia-southeast1`
- `europe-north1`
- `europe-west2`
- `europe-west3`
- `europe-west6`
- `northamerica-northeast1`
- `southamerica-east1`
You can increase resource quotas from the GCP console, but you might need to file a support ticket. Be sure to plan your cluster size early so that you can allow time to resolve the support ticket before you install your OpenShift Container Platform cluster.

4.11.4.5. Creating a service account in GCP

OpenShift Container Platform requires a Google Cloud Platform (GCP) service account that provides authentication and authorization to access data in the Google APIs. If you do not have an existing IAM service account that contains the required roles in your project, you must create one.

Prerequisites

- You created a project to host your cluster.

Procedure

1. Create a service account in the project that you use to host your OpenShift Container Platform cluster. See Creating a service account in the GCP documentation.

2. Grant the service account the appropriate permissions. You can either grant the individual permissions that follow or assign the Owner role to it. See Granting roles to a service account for specific resources.

   NOTE

   While making the service account an owner of the project is the easiest way to gain the required permissions, it means that service account has complete control over the project. You must determine if the risk that comes from offering that power is acceptable.

3. Create the service account key in JSON format. See Creating service account keys in the GCP documentation.

   The service account key is required to create a cluster.

4.11.4.5.1. Required GCP permissions

When you attach the Owner role to the service account that you create, you grant that service account all permissions, including those that are required to install OpenShift Container Platform. To deploy an OpenShift Container Platform cluster, the service account requires the following permissions. If you deploy your cluster into an existing VPC, the service account does not require certain networking permissions, which are noted in the following lists:

Required roles for the installation program

- Compute Admin
- Security Admin
- Service Account Admin
- Service Account User
- Storage Admin
Required roles for creating network resources during installation

- DNS Administrator

Required roles for user-provisioned GCP infrastructure

- Deployment Manager Editor
- Service Account Key Admin

Optional roles

For the cluster to create new limited credentials for its Operators, add the following role:

- Service Account Key Admin

The roles are applied to the service accounts that the control plane and compute machines use:

**Table 4.47. GCP service account permissions**

<table>
<thead>
<tr>
<th>Account</th>
<th>Roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Plane</td>
<td>roles/compute.instanceAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.networkAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/compute.securityAdmin</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
<tr>
<td></td>
<td>roles/iam.serviceAccountUser</td>
</tr>
<tr>
<td>Compute</td>
<td>roles/compute.viewer</td>
</tr>
<tr>
<td></td>
<td>roles/storage.admin</td>
</tr>
</tbody>
</table>

4.11.4.6. Supported GCP regions

You can deploy an OpenShift Container Platform cluster to the following Google Cloud Platform (GCP) regions:

- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-southeast1** (Jurong West, Singapore)
4.11.4.7. Installing and configuring CLI tools for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must install and configure the CLI tools for GCP.

**Prerequisites**

- You created a project to host your cluster.
- You created a service account and granted it the required permissions.

**Procedure**

1. Install the following binaries in $PATH:
   - `gcloud`
   - `gsutil`

   See Install the latest Cloud SDK version in the GCP documentation.

2. Authenticate using the `gcloud` tool with your configured service account. See Authorizing with a service account in the GCP documentation.
4.11.5. Creating the installation files for GCP

To install OpenShift Container Platform on Google Cloud Platform (GCP) using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the `install-config.yaml` file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate `/var` partition during the preparation phases of installation.

4.11.5.1. Optional: Creating a separate `/var` partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.

- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.

- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

**IMPORTANT**

If you follow the steps to create a separate `/var` partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ```

**Example output**

```
? SSH Public Key ...
```
3. Optional: Confirm that the installation program created manifests in the `clusterconfig/openshift` directory:

   ```
   $ ls $HOME/clusterconfig/openshift/
   ```

   **Example output**

   99_kubeadmin-password-secret.yaml  
   99_openshift-cluster-api_master-machines-0.yaml  
   99_openshift-cluster-api_master-machines-1.yaml  
   99_openshift-cluster-api_master-machines-2.yaml

   ...

4. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
       machineconfiguration.openshift.io/role: worker
   name: 98-var-partition
   spec:
     config:
       ignition:
         version: 3.1.0
     storage:
       disks:
         - device: /dev/<device_name>  
         partitions:
           - label: var
             startMiB: <partition_start_offset>  
             sizeMiB: <partition_size>  
         filesystems:
           - device: /dev/disk/by-partlabel/var
             path: /var
             format: xfs
     systemd:
       units:
         - name: var.mount  
           enabled: true
           contents: |
           [Unit]
           Before=local-fs.target
           [Mount]
           What=/dev/disk/by-partlabel/var
           Where=/var
   ```
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-lib-containers.mount`.

The `prjquota` mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 4.11.5.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Google Cloud Platform (GCP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

- Have the `imageContentSources` values that were generated during mirror registry creation.

- Obtain the contents of the certificate for your mirror registry.

**Procedure**

1. Create the `install-config.yaml` file.
a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `gcp` as the platform to target.

iii. If you have not configured the service account key for your GCP account on your computer, you must obtain it from GCP and paste the contents of the file or enter the absolute path to the file.

iv. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

v. Select the region to deploy the cluster to.

vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

vii. Enter a descriptive name for your cluster.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Edit the `install-config.yaml` file to provide the additional information that is required for an installation in a restricted network.

a. Update the `pullSecret` value to contain the authentication information for your registry:

```
pullSecret: "{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>"},"email": "you@example.com"}}}
```
For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

b. Add the `additionalTrustBundle` parameter and value.

```
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
    ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
    -----END CERTIFICATE-----
```

The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

c. Define the network and subnets for the VPC to install the cluster in under the parent `platform.gcp` field:

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

For `platform.gcp.network`, specify the name for the existing Google VPC. For `platform.gcp.controlPlaneSubnet` and `platform.gcp.computeSubnet`, specify the existing subnets to deploy the control plane machines and compute machines, respectively.

d. Add the image content resources, which look like this excerpt:

```
imageContentSources:
  - mirrors:
    - `<mirror_host_name>:5000/<repo_name>/release
      source: quay.example.com/openshift-release-dev/ocp-release
    - mirrors:
      - `<mirror_host_name>:5000/<repo_name>/release
       source: registry.example.com/ocp/release
```

To complete these values, use the `imageContentSources` that you recorded during mirror registry creation.

3. Make any other modifications to the `install-config.yaml` file that you require. You can find more information about the available parameters in the `Installation configuration parameters` section.

4. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

4.11.5.3. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
  noProxy: example.com  # 3
additionalTrustBundle: |
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
...
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
4. If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents.
specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE
The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE
Only the Proxy object named cluster is supported, and no additional proxies can be created.

4.11.5.4. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.

- You created the install-config.yaml installation configuration file.

Procedure
1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory>
```

**Example output**

```
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift
```

For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Optional: If you do not want the cluster to provision compute machines, remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yaml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yaml` file.

b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

c. Save and exit the file.

5. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the `privateZone` and `publicZone` sections from the `<installation_directory>/manifests/cluster-dns-02-config.yaml` DNS configuration file:

```json
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 
    id: mycluster-100419-private-zone
```

```python
$ ./openshift-install create manifests --dir <installation_directory>
```

INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift
To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

Additional resources

- Optional: Adding the ingress DNS records

### 4.11.6. Exporting common variables

#### 4.11.6.1. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The infrastructure name is also used to locate the appropriate GCP resources during an OpenShift Container Platform installation. The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

**Procedure**

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6
```

The output of this command is your cluster name and a random string.

4.11.6.2. Exporting common variables for Deployment Manager templates

You must export a common set of variables that are used with the provided Deployment Manager templates used to assist in completing a user-provided infrastructure install on Google Cloud Platform (GCP).

**NOTE**

Specific Deployment Manager templates can also require additional exported variables, which are detailed in their related procedures.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
- Generate the Ignition config files for your cluster.
- Install the `jq` package.

**Procedure**

1. Export the following common variables to be used by the provided Deployment Manager templates:

```
$ jq -r .infraID <installation_directory>/metadata.json
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/19'
$ export WORKER_SUBNET_CIDR='10.0.32.0/19'
```

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

```
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
4.11.7. Creating a VPC in GCP

You must create a VPC in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. You can customize the VPC to meet your requirements. One way to create the VPC is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.

**Procedure**

1. Copy the template from the Deployment Manager template for the VPC section of this topic and save it as `01_vpc.py` on your computer. This template describes the VPC that your cluster requires.

2. Create a `01_vpc.yaml` resource definition file:

```bash
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}'
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}'
EOF
```

1. `infra_id` is the `INFRA_ID` infrastructure name from the extraction step.
2. `region` is the region to deploy the cluster into, for example `us-central1`.
3. `master_subnet_cidr` is the CIDR for the master subnet, for example `10.0.0.0/19`.
4. `worker_subnet_cidr` is the CIDR for the worker subnet, for example `10.0.32.0/19`.

3. Create the deployment by using the `gcloud` CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```
4.11.7.1. Deployment Manager template for the VPC

You can use the following Deployment Manager template to deploy the VPC that you need for your OpenShift Container Platform cluster:

Example 4.19. 01_vpc.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-network',
         'type': 'compute.v1.network',
         'properties': {
             'region': context.properties['region'],
             'autoCreateSubnetworks': False
         }},
        {'name': context.properties['infra_id'] + '-master-subnet',
         'type': 'compute.v1.subnetwork',
         'properties': {
             'region': context.properties['region'],
             'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
             'ipCidrRange': context.properties['master_subnet_cidr']
         }},
        {'name': context.properties['infra_id'] + '-worker-subnet',
         'type': 'compute.v1.subnetwork',
         'properties': {
             'region': context.properties['region'],
             'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
             'ipCidrRange': context.properties['worker_subnet_cidr']
         }},
        {'name': context.properties['infra_id'] + '-router',
         'type': 'compute.v1.router',
         'properties': {
             'region': context.properties['region'],
             'network': '${ref.' + context.properties['infra_id'] + '-network.selfLink}',
             'nats': [{
                 'name': context.properties['infra_id'] + '-nat-master',
                 'natIpAllocateOption': 'AUTO_ONLY',
                 'minPortsPerVm': 7168,
                 'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                 'subnetworks': [{
                     'name': '${ref.' + context.properties['infra_id'] + '-master-subnet.selfLink}',
                     'sourceIpRangesToNat': ['ALL_IP_RANGES']
                 }],
                 'natIpAllocateOption': 'AUTO_ONLY',
                 'minPortsPerVm': 512,
                 'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                 'subnetworks': [{
                     'name': '${ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink}',
                     'sourceIpRangesToNat': ['ALL_IP_RANGES']
                 }]}}
    ]
```
4.11.8. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 4.48. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 4.49. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 4.50. Control plane machines to control plane machines
### Network topology requirements

The infrastructure that you provision for your cluster must meet the following network topology requirements.

### Load balancers

Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   IMPORTANT
   
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

### Table 4.51. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer:** Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

### 4.11.9. Creating load balancers in GCP

You must configure load balancers in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.
NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the Deployment Manager template for the internal load balancer section of this topic and save it as 02_lb_int.py on your computer. This template describes the internal load balancing objects that your cluster requires.

2. For an external cluster, also copy the template from the Deployment Manager template for the external load balancer section of this topic and save it as 02_lb_ext.py on your computer. This template describes the external load balancing objects that your cluster requires.

3. Export the variables that the deployment template uses:
   
a. Export the cluster network location:

   ```
   $ export CLUSTER_NETWORK=('gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink')
   ```

   b. Export the control plane subnet location:

   ```
   $ export CONTROL_SUBNET=('gcloud compute networks subnets describe $(INFRA_ID)-master-subnet --region=${REGION} --format json | jq -r .selfLink')
   ```

   c. Export the three zones that the cluster uses:

   ```
   $ export ZONE_0=('gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9')
   $ export ZONE_1=('gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9')
   $ export ZONE_2=('gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9')
   ```

4. Create a 02_infra.yaml resource definition file:

   ```
   $ cat <<EOF >02_infra.yaml
   imports:
   - path: 02_lb_ext.py
   - path: 02_lb_int.py
   EOF
   ```
resources:
- name: cluster-lb-ext
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}'
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: '${ZONE_0}', '${ZONE_1}', '${ZONE_2}'
EOF

1. Required only when deploying an external cluster.
2. `infra_id` is the `INFRA_ID` infrastructure name from the extraction step.
3. `region` is the region to deploy the cluster into, for example `us-central1`.
4. `control_subnet` is the URI to the control subnet.
5. `zones` are the zones to deploy the control plane instances into, like `us-east1-b, us-east1-c, and us-east1-d`.

5. Create the deployment by using the `gcloud` CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. Export the cluster IP address:

```
$ export CLUSTER_IP=`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --region=${REGION} --format json | jq -r .address`
```

7. For an external cluster, also export the cluster public IP address:

```
$ export CLUSTER_PUBLIC_IP=`gcloud compute addresses describe ${INFRA_ID}-cluster-public-ip --region=${REGION} --format json | jq -r .address`
```

4.11.9.1. Deployment Manager template for the external load balancer

You can use the following Deployment Manager template to deploy the external load balancer that you need for your OpenShift Container Platform cluster:

**Example 4.20. 02_lb_ext.py Deployment Manager template**

```python
def GenerateConfig(context):
```
You can use the following Deployment Manager template to deploy the internal load balancer that you need for your OpenShift Container Platform cluster:

Example 4.21. 02_lb_int.py Deployment Manager template

def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '${ref.' + context.properties['infra_id'] + '-master-' + zone + '-instance-group' + '.selfLink}'
        })

    resources = [
        {'name': context.properties['infra_id'] + '-cluster-ip',
         'type': 'compute.v1.address',
         'properties': {
            'region': context.properties['region']
        }},
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
        # probe for kube-apiserver
        {'name': context.properties['infra_id'] + '-api-http-health-check',
         'type': 'compute.v1.httpHealthCheck',
         'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }},
        {'name': context.properties['infra_id'] + '-api-target-pool',
         'type': 'compute.v1.targetPool',
         'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }},
        {'name': context.properties['infra_id'] + '-api-forwarding-rule',
         'type': 'compute.v1.forwardingRule',
         'properties': {
            'region': context.properties['region'],
            'IPAddress': '${ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink}',
            'target': '${ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink}',
            'portRange': '6443'
        }]
    return {'resources': resources}
for zone in context.properties['zones']:
  resources.append(
    {
      'name': context.properties['infra_id'] + '-master-' + zone + '-instance-group',
      'type': 'compute.v1.instanceGroup',
      'properties': {
        'namedPorts': [
          {
            'name': 'ignition',
            'port': 22623
          },
          {
            'name': 'https',
            'port': 6443
          }
        ]
    }
  )
}]

# Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check probe for kube-apiserver

'name': context.properties['infra_id'] + '-api-internal-health-check',
'type': 'compute.v1.healthCheck',
'properties': {
  'httpsHealthCheck': {
    'port': 6443,
    'requestPath': '/readyz'
  },
  'type': "HTTPS"
}
}, {
  'name': context.properties['infra_id'] + '-api-internal-backend-service',
  'type': 'compute.v1.regionBackendService',
  'properties': {
    'backends': backends,
    'healthChecks': ['$ref.' + context.properties['infra_id'] + '-api-internal-health-check.selfLink'],
    'loadBalancingScheme': 'INTERNAL',
    'region': context.properties['region'],
    'protocol': 'TCP',
    'timeoutSec': 120
  }
}, {
  'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
  'type': 'compute.v1.forwardingRule',
  'properties': {
    'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-service.selfLink)',
    'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
    'loadBalancingScheme': 'INTERNAL',
    'ports': ['6443', '22623'],
    'region': context.properties['region'],
    'subnetwork': context.properties['control_subnet']
  }
})
4.11.10. Creating a private DNS zone in GCP

You must configure a private DNS zone in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create this component is to modify the provided Deployment Manager template.

NOTE

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

Procedure

1. Copy the template from the Deployment Manager template for the private DNS section of this topic and save it as 02_dns.py on your computer. This template describes the private DNS objects that your cluster requires.

2. Create a 02_dns.yaml resource definition file:

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py
resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}'
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}'
    cluster_network: '${CLUSTER_NETWORK}'
EOF
```
1. `infra_id` is the INFRA_ID infrastructure name from the extraction step.

2. `cluster_domain` is the domain for the cluster, for example `openshift.example.com`.

3. `cluster_network` is the selfLink URL to the cluster network.

3. Create the deployment by using the `gcloud` CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
   ```

4. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

   a. Add the internal DNS entries:

   ```
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api.$(CLUSTER_NAME).$(BASE_DOMAIN). --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-int.$(CLUSTER_NAME).$(BASE_DOMAIN). --ttl 60 --type A --zone ${INFRA_ID}-private-zone
   $ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
   ```

   b. For an external cluster, also add the external DNS entries:

   ```
   $ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
   $ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
   $ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name api.$(CLUSTER_NAME).$(BASE_DOMAIN). --ttl 60 --type A --zone $(BASE_DOMAIN_ZONE_NAME)
   $ gcloud dns record-sets transaction execute --zone $(BASE_DOMAIN_ZONE_NAME)
   ```

4.11.10.1. Deployment Manager template for the private DNS

You can use the following Deployment Manager template to deploy the private DNS that you need for your OpenShift Container Platform cluster:

```python
Example 4.22. 02_dns.py Deployment Manager template

def GenerateConfig(context):
    
    resources = [
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [
                    {'networkUrl': context.properties['cluster_network']}
                ]
            }
        }
    ]
```
4.11.1. Creating firewall rules in GCP

You must create firewall rules in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**

1. Copy the template from the Deployment Manager template for firewall rules section of this topic and save it as `03_firewall.py` on your computer. This template describes the security groups that your cluster requires.

2. Create a `03_firewall.yaml` resource definition file:

   ```yaml
   $ cat <<EOF >03_firewall.yaml
   imports:
     - path: 03_firewall.py
   resources:
     - name: cluster-firewall
       type: 03_firewall.py
       properties:
         allowed_external_cidr: '0.0.0.0/0' ①
         infra_id: '${INFRA_ID}' ②
         cluster_network: '${CLUSTER_NETWORK}' ③
         network_cidr: '${NETWORK_CIDR}' ④
   EOF
   
   ① *allowed_external_cidr* is the CIDR range that can access the cluster API and SSH to the bootstrap host. For an internal cluster, set this value to `$(NETWORK_CIDR)`.
   
   ② *infra_id* is the *INFRA_ID* infrastructure name from the extraction step.
3. **cluster_network** is the **selfLink** URL to the cluster network.

4. **network_cidr** is the CIDR of the VPC network, for example **10.0.0.0/16**.

3. Create the deployment by using the **gcloud** CLI:

   ```bash
   $ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config 03_firewall.yaml
   ```

### 4.11.11.1. Deployment Manager template for firewall rules

You can use the following Deployment Manager template to deploy the firewall rules that you need for your OpenShift Container Platform cluster:

#### Example 4.23. 03_firewall.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {
            'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
            'type': 'compute.v1.firewall',
            'properties': {
                'network': context.properties['cluster_network'],
                'allowed': [{
                    'IPProtocol': 'tcp',
                    'ports': ['22']
                }],
                'sourceRanges': [context.properties['allowed_external_cidr']],
                'targetTags': [context.properties['infra_id'] + '-bootstrap']
            }
        },
        {
            'name': context.properties['infra_id'] + '-api',
            'type': 'compute.v1.firewall',
            'properties': {
                'network': context.properties['cluster_network'],
                'allowed': [{
                    'IPProtocol': 'tcp',
                    'ports': ['6443']
                }],
                'sourceRanges': [context.properties['allowed_external_cidr']],
                'targetTags': [context.properties['infra_id'] + '-master']
            }
        },
        {
            'name': context.properties['infra_id'] + '-health-checks',
            'type': 'compute.v1.firewall',
            'properties': {
                'network': context.properties['cluster_network'],
                'allowed': [{
                    'IPProtocol': 'tcp',
                    'ports': ['6080', '6443', '22624']
                }],
                'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
                'targetTags': [context.properties['infra_id'] + '-master']
            }
        }
    ]
```
}, {
  'name': context.properties['infra_id'] + '-etcd',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['2379-2380']
    }],
    'sourceTags': [context.properties['infra_id'] + '-master'],
    'targetTags': [context.properties['infra_id'] + '-master']
  }
}, {
  'name': context.properties['infra_id'] + '-control-plane',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['10257']
    },{
      'IPProtocol': 'tcp',
      'ports': ['10259']
    },{
      'IPProtocol': 'tcp',
      'ports': ['22623']
    }],
    'sourceTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ],
    'targetTags': [context.properties['infra_id'] + '-master']
  }
}, {
  'name': context.properties['infra_id'] + '-internal-network',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'icmp'
    }, {
      'IPProtocol': 'tcp',
      'ports': ['22']
    }],
    'sourceRanges': [context.properties['network_cidr']],
    'targetTags': [
      context.properties['infra_id'] + '-master',
      context.properties['infra_id'] + '-worker'
    ]
  }
}, {
  'name': context.properties['infra_id'] + '-internal-cluster',
  'type': 'compute.v1.firewall',
  'properties': {
    'network': context.properties['cluster_network'],
    'allowed': [{
      'IPProtocol': 'tcp',
      'ports': ['22']
    }]
  }
}
4.11.12. Creating IAM roles in GCP

You must create IAM roles in Google Cloud Platform (GCP) for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your GCP infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

**Prerequisites**

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.

**Procedure**
1. Copy the template from the Deployment Manager template for IAM roles section of this topic and save it as 03_iam.py on your computer. This template describes the IAM roles that your cluster requires.

2. Create a 03_iam.yaml resource definition file:

```bash
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}'
EOF
```

`infra_id` is the INFRA_ID infrastructure name from the extraction step.

3. Create the deployment by using the gcloud CLI:

```bash
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. Export the variable for the master service account:

```bash
$ export MASTER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

5. Export the variable for the worker service account:

```bash
$ export WORKER_SERVICE_ACCOUNT=$(gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email')
```

6. Export the variable for the subnet that hosts the compute machines:

```bash
$ export COMPUTE_SUBNET=$(gcloud compute networks subnets describe ${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink)
```

7. The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```bash
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
```
8. Create a service account key and store it locally for later use:

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-account=${MASTER_SERVICEACCOUNT}
```

### 4.11.12.1. Deployment Manager template for IAM roles

You can use the following Deployment Manager template to deploy the IAM roles that you need for your OpenShift Container Platform cluster:

#### Example 4.24. 03_iam.py Deployment Manager template

```python
def GenerateConfig(context):
    resources = [
        {'name': context.properties['infra_id'] + '-master-node-sa',
         'type': 'iam.v1.serviceAccount',
         'properties': {
             'accountId': context.properties['infra_id'] + '-m',
             'displayName': context.properties['infra_id'] + '-master-node'
         }},
        {'name': context.properties['infra_id'] + '-worker-node-sa',
         'type': 'iam.v1.serviceAccount',
         'properties': {
             'accountId': context.properties['infra_id'] + '-w',
             'displayName': context.properties['infra_id'] + '-worker-node'
         }},
    ]
    return {'resources': resources}
```

### 4.11.13. Creating the RHCOS cluster image for the GCP infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Google Cloud Platform (GCP) for your OpenShift Container Platform nodes.

#### Procedure

1. Obtain the RHCOS image from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.
The file name contains the OpenShift Container Platform version number in the format \texttt{rhcos-<version>-<arch>-gcp.<arch>.tar.gz}.

2. Create the Google storage bucket:
   
   \begin{verbatim}
   $ gsutil mb gs://<bucket_name>
   \end{verbatim}

3. Upload the RHCOS image to the Google storage bucket:
   
   \begin{verbatim}
   $ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz gs://<bucket_name>
   \end{verbatim}

4. Export the uploaded RHCOS image location as a variable:
   
   \begin{verbatim}
   $ export IMAGE_SOURCE=`gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz`
   \end{verbatim}

5. Create the cluster image:
   
   \begin{verbatim}
   $ gcloud compute images create "${INFRA_ID}-rhcos-image" --source-uri="${IMAGE_SOURCE}"
   \end{verbatim}

4.11.14. Creating the bootstrap machine in GCP

You must create the bootstrap machine in Google Cloud Platform (GCP) to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Deployment Manager template.

\begin{quote}
\textbf{NOTE}

If you do not use the provided Deployment Manager template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.
\end{quote}

Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Ensure pyOpenSSL is installed.

Procedure

1. Copy the template from the Deployment Manager template for the bootstrap machine section of this topic and save it as \texttt{04_bootstrap.py} on your computer. This template describes the bootstrap machine that your cluster requires.
2. Export the location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that the installation program requires:

   ```bash
   $ export CLUSTER_IMAGE=`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`
   ```

3. Create a bucket and upload the `bootstrap.ign` file:

   ```bash
   $ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
   $ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
   ```

4. Create a signed URL for the bootstrap instance to use to access the Ignition config. Export the URL from the output as a variable:

   ```bash
   $ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep '^gs:' | awk '{print $5}'`
   ```

5. Create a `04_bootstrap.yaml` resource definition file:

   ```yaml
   $ cat <<EOF >04_bootstrap.yaml
   imports:
   - path: 04_bootstrap.py
   resources:
   - name: cluster-bootstrap
     type: 04_bootstrap.py
     properties:
       infra_id: '${INFRA_ID}'
       region: '${REGION}'
       zone: '${ZONE_0}'
       cluster_network: '${CLUSTER_NETWORK}'
       control_subnet: '${CONTROL_SUBNET}'
       image: '${CLUSTER_IMAGE}'
       machine_type: 'n1-standard-4'
       root_volume_size: '128'
       bootstrap_ign: '${BOOTSTRAP_IGN}'
   EOF
   ```

   - **infra_id** is the `INFRA_ID` infrastructure name from the extraction step.
   - **region** is the region to deploy the cluster into, for example `us-central1`.
   - **zone** is the zone to deploy the bootstrap instance into, for example `us-central1-b`.
   - **cluster_network** is the `selfLink` URL to the cluster network.
   - **control_subnet** is the `selfLink` URL to the control subnet.
   - **image** is the `selfLink` URL to the RHCOS image.
   - **machine_type** is the machine type of the instance, for example `n1-standard-4`. 

---

974
8. **root_volume_size** is the boot disk size for the bootstrap machine.

9. **bootstrap_url** is the URL output when creating a signed URL.

6. Create the deployment by using the `gcloud` CLI:

   ```bash
   $ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config 04_bootstrap.yaml
   ```

7. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the bootstrap machine manually.

   a. Add the bootstrap instance to the internal load balancer instance group:

   ```bash
   $ gcloud compute instance-groups unmanaged add-instances \
   ${INFRA_ID}-bootstrap-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
   ```

   b. Add the bootstrap instance group to the internal load balancer backend service:

   ```bash
   $ gcloud compute backend-services add-backend \
   ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
   ```

4.11.14.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Manager template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

**Example 4.25. 04_bootstrap.py Deployment Manager template**

```python
def GenerateConfig(context):

    resources = [
        {'name': context.properties['infra_id'] + '-bootstrap-public-ip',
         'type': 'compute.v1.address',
         'properties': {
             'region': context.properties['region']
         }
        },
        {'name': context.properties['infra_id'] + '-bootstrap',
         'type': 'compute.v1.instance',
         'properties': {
             'disks': [{
                 'autoDelete': True,
                 'boot': True,
                 'initializeParams': {
                     'diskSizeGb': context.properties['root_volume_size'],
                     'sourceImage': context.properties['image']
                 }
             }],
             'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' + context.properties['machine_type'],
         }
    ]
```
4.11.15. Creating the control plane machines in GCP

You must create the control plane machines in Google Cloud Platform (GCP) for your cluster to use. One way to create these machines is to modify the provided Deployment Manager template.

**NOTE**

If you do not use the provided Deployment Manager template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.
Prerequisites

- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.

Procedure

1. Copy the template from the Deployment Manager template for control plane machines section of this topic and save it as `05_control_plane.py` on your computer. This template describes the control plane machines that your cluster requires.

2. Export the following variable required by the resource definition:

   ```
   $ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
   ```

3. Create a `05_control_plane.yaml` resource definition file:

   ```
   $ cat <<EOF >05_control_plane.yaml
   imports:
   - path: 05_control_plane.py

   resources:
   - name: cluster-control-plane
     type: 05_control_plane.py
     properties:
       infra_id: '${INFRA_ID}'
       zones: 
         - '${ZONE_0}'
         - '${ZONE_1}'
         - '${ZONE_2}'
       control_subnet: '${CONTROL_SUBNET}'
       image: '${CLUSTER_IMAGE}'
       machine_type: 'n1-standard-4'
       root_volume_size: '128'
       service_account_email: '${MASTER_SERVICE_ACCOUNT}'
       ignition: '${MASTER_IGNITION}'
   EOF
   ```

   `infra_id` is the `INFRA_ID` infrastructure name from the extraction step.

   `zones` are the zones to deploy the control plane instances into, for example `us-central1-a`, `us-central1-b`, and `us-central1-c`.

   `control_subnet` is the `selfLink` URL to the control subnet.
4. *image* is the selfLink URL to the RH COS image.

5. *machine_type* is the machine type of the instance, for example *n1-standard-4*.

6. *service_account_email* is the email address for the master service account that you created.

7. *ignition* is the contents of the *master.ign* file.

4. Create the deployment by using the *gcloud* CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
   05_control_plane.yaml
   ```

5. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually.

   - Run the following commands to add the control plane machines to the appropriate instance groups:

     ```
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-          
     ${ZONE_0}-instance-group --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-          
     ${ZONE_1}-instance-group --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
     $ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-          
     ${ZONE_2}-instance-group --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
     ```

   - For an external cluster, you must also run the following commands to add the control plane machines to the target pools:

     ```
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-        
     zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-        
     zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
     $ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-        
     zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
     ```

4.11.15.1. Deployment Manager template for control plane machines

You can use the following Deployment Manager template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

**Example 4.26. 05_control_plane.py Deployment Manager template**

```python
def GenerateConfig(context):
    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
```
'diskSizeGb': context.properties['root_volume_size'],
'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
'sourceImage': context.properties['image']
},
'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
'metadata': {
'items': {
  'key': 'user-data',
  'value': context.properties['ignition']
}
},
'networkInterfaces': [{
  'subnetwork': context.properties['control_subnet']
}],
'serviceAccounts': [{
  'email': context.properties['service_account_email'],
  'scopes': ['https://www.googleapis.com/auth/cloud-platform']
}],
'tags': {
'items': [context.properties['infra_id'] + '-master',
  ]
},
'zone': context.properties['zones'][0]
},
{name: context.properties['infra_id'] + '-master-1',
'type': 'compute.v1.instance',
'properties': {
'disks': [{
  'autoDelete': True,
  'boot': True,
  'initializeParams': {
    'diskSizeGb': context.properties['root_volume_size'],
    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
    'sourceImage': context.properties['image']
  }
}],
'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
'metadata': {
'items': {
  'key': 'user-data',
  'value': context.properties['ignition']
}
},
'networkInterfaces': [{
  'subnetwork': context.properties['control_subnet']
}],
'serviceAccounts': [{
  'email': context.properties['service_account_email'],
  'scopes': ['https://www.googleapis.com/auth/cloud-platform']
}],
'tags': {
4.11.16. Wait for bootstrap completion and remove bootstrap resources in GCP

After you create all of the required infrastructure in Google Cloud Platform (GCP), wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Configure a GCP account.
 Generate the Ignition config files for your cluster.

 Create and configure a VPC and associated subnets in GCP.

 Create and configure networking and load balancers in GCP.

 Create control plane and compute roles.

 Create the bootstrap machine.

 Create the control plane machines.

 Procedure

 1. Change to the directory that contains the installation program and run the following command:

    ```bash
    $ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \  
    --log-level info
    ```

   1  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   2  To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   If the command exits without a **FATAL** warning, your production control plane has initialized.

 2. Delete the bootstrap resources:

    ```bash
    $ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-instance-group --instance-group-zone=${ZONE_0}
    $ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
    $ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
    $ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
    ```

4.11.17. Creating additional worker machines in GCP

You can create worker machines in Google Cloud Platform (GCP) for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Deployment Manager template. Additional instances can be launched by including additional resources of type `06_worker.py` in the file.

**NOTE**

If you do not use the provided Deployment Manager template to create your worker machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites
- Configure a GCP account.
- Generate the Ignition config files for your cluster.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.

Procedure

1. Copy the template from the Deployment Manager template for worker machines section of this topic and save it as 06_worker.py on your computer. This template describes the worker machines that your cluster requires.

2. Export the variables that the resource definition uses.
   a. Export the subnet that hosts the compute machines:

   ```bash
   $ export COMPUTE_SUBNET=`gcloud compute networks subnets describe $(INFRA_ID)-worker-subnet --region=${REGION} --format json | jq -r .selfLink`
   ```

   b. Export the email address for your service account:

   ```bash
   $ export WORKER_SERVICE_ACCOUNT=`gcloud iam service-accounts list --filter "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'
   ```

   c. Export the location of the compute machine Ignition config file:

   ```bash
   $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
   ```

3. Create a 06_worker.yaml resource definition file:

   ```bash
   $ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py
resources:
- name: 'worker-0'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}'
    zone: '${ZONE_0}'
    compute_subnet: '${COMPUTE_SUBNET}'
    image: '${CLUSTER_IMAGE}'
    machine_type: 'n1-standard-4'
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}'
    ignition: '${WORKER_IGNITION}'
EOF
   ```
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}'
    zone: '${ZONE_1}'
    compute_subnet: '${COMPUTE_SUBNET}''
    image: '${CLUSTER_IMAGE}''
    machine_type: 'n1-standard-4'
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}''
    ignition: '${WORKER_IGNITION}''
EOF

1. **name** is the name of the worker machine, for example **worker-0**.

2. **infra_id** is the **INFRA_ID** infrastructure name from the extraction step.

3. **zone** is the zone to deploy the worker machine into, for example **us-central1-a**.

4. **compute_subnet** is the **selfLink** URL to the compute subnet.

5. **image** is the **selfLink** URL to the RHCOS image.

6. **machine_type** is the machine type of the instance, for example **n1-standard-4**.

7. **service_account_email** is the email address for the worker service account that you created.

8. **ignition** is the contents of the **worker.ign** file.

4. Optional: If you want to launch additional instances, include additional resources of type **06_worker.py** in your **06_worker.yaml** resource definition file.

5. Create the deployment by using the **gcloud** CLI:

   ```
   $ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config 06_worker.yaml
   ```

4.11.17.1. **Deployment Manager template for worker machines**

You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

**Example 4.27. 06_worker.py Deployment Manager template**

```python
    def GenerateConfig(context):
        resources = {
            'name': context.properties['infra_id'] + '-' + context.env['name'],
            'type': 'compute.v1.instance',
            'properties': {
                'disks': [{
                    'autoDelete': True,
                    'boot': True,
```
You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadm credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

**Example output**

```
system:admin
```

### 4.11.19. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

```
$ oc patch OperatorHub cluster --type json \
- p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the **Administration** → **Cluster Settings** → **Global Configuration** → **OperatorHub** page, click the **Sources** tab, where you can create, delete, disable, and enable individual sources.

### 4.11.20. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

**Example output**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

The output lists all of the machines that you created.
NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the Pending or Approved status for each machine that you added to the cluster:

   $ oc get csr

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
</tbody>
</table>

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in Pending status, approve the CSRs for your cluster machines:

   NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the machine-approver if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the oc exec, oc rsh, and oc logs commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the system:node or system:admin groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  $ oc adm certificate approve <csr_name>
<csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs --no-run-if-empty oc adm certificate approve
```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57tv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>
```

- <csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>
It can take a few minutes after approval of the server CSRs for the machines to transition to the Ready status.

Additional information

- For more information on CSRs, see Certificate Signing Requests.

### 4.11.21. Optional: Adding the ingress DNS records

If you removed the DNS zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the ingress load balancer. You can create either a wildcard `*.apps.{baseDomain}` or specific records. You can use A, CNAME, and other records per your requirements.

**Prerequisites**

- Configure a GCP account.
- Remove the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs.
- Create and configure a VPC and associated subnets in GCP.
- Create and configure networking and load balancers in GCP.
- Create control plane and compute roles.
- Create the bootstrap machine.
- Create the control plane machines.
- Create the worker machines.

**Procedure**

1. Wait for the Ingress router to create a load balancer and populate the EXTERNAL-IP field:

   ```
   $ oc -n openshift-ingress get service router-default
   ```

   **Example output**

   ```
   NAME             TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)                      AGE
   ```

2. Add the A record to your zones:

   - To use A records:
i. Export the variable for the router IP address:

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk '{print $4}'`
```

ii. Add the A record to the private zones:

```
$ if [-f transaction.yaml]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name \\
*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone \\
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

iii. For an external cluster, also add the A record to the public zones:

```
$ if [-f transaction.yaml]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name \\
*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone \\
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone \\
${BASE_DOMAIN_ZONE_NAME}
```

- To add explicit domains instead of using a wildcard, create entries for each of the cluster’s current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host} \\
{"\n"}{end}{end}' routes
```

Example output

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
grafana-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

4.11.22. Completing a GCP installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Google Cloud Platform (GCP) user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned GCP infrastructure.

- Install the `oc` CLI and log in.

Procedure
1. Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Observe the running state of your cluster.

   a. Run the following command to view the current cluster version and status:

```
$ oc get clusterversion
```

**Example output**

```
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version             False       True          24m     Working towards 4.5.4: 99% complete
```

b. Run the following command to view the Operators managed on the control plane by the Cluster Version Operator (CVO):

```
$ oc get clusteroperators
```

**Example output**

```
NAME                                      VERSION   AVAILABLE   PROGRESSING   DEGRADED SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
```
c. Run the following command to view your cluster pods:

```
$ oc get pods --all-namespaces
```

**Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-111.us-east-2.com</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>2.compute.internal</td>
<td>1/1 Running 0 35m</td>
</tr>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-239.us-east-2.com</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>2.compute.internal</td>
<td>1/1 Running 0 37m</td>
</tr>
<tr>
<td>kube-system</td>
<td>etcd-member-ip-10-0-3-24.us-east-2.com</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>2.compute.internal</td>
<td>1/1 Running 0 35m</td>
</tr>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-6d667f4f4f-h7t2t</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>1/1 Running 0 30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-fxkvv</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>1/1 Running 0 29m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-q85nm</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>1/1 Running 0 29m</td>
</tr>
<tr>
<td>openshift-service-ca-operator</td>
<td>openshift-service-ca-operator-66ff6dc6cd-9r257</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>1/1 Running 0 37m</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>apiserver-cabundle-injector-695b6bcbc-cl5hm</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>1/1 Running 0 35m</td>
</tr>
<tr>
<td>openshift-service-ca</td>
<td>configmap-cabundle-injector-8498544d7-</td>
</tr>
</tbody>
</table>

---

CHAPTER 4. INSTALLING ON GCP
When the current cluster version is **AVAILABLE**, the installation is complete.

### 4.11.23. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service.

### 4.11.24. Next steps

- Customize your cluster.

- Configure image streams for the Cluster Samples Operator and the `must-gather` tool.

- Learn how to [use Operator Lifecycle Manager (OLM) on restricted networks](#).

- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.

- If necessary, you can opt out of remote health reporting.

### 4.12. UNINSTALLING A CLUSTER ON GCP

You can remove a cluster that you deployed to Google Cloud Platform (GCP).

#### 4.12.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

**NOTE**

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access. For example, some Google Cloud resources require IAM permissions in shared VPC host projects, or there might be unused health checks that must be deleted.

**Prerequisites**

---

<table>
<thead>
<tr>
<th>25qn6</th>
<th>1/1</th>
<th>Running</th>
<th>0</th>
<th>35m</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-service-ca</td>
<td>service-serving-cert-signer-6445fc9c6-wqdqn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>35m</td>
<td></td>
</tr>
<tr>
<td>openshift-service-catalog-apiserver-operator</td>
<td>openshift-service-catalog-apiserver-operator-549f44668b-b5q2w</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>32m</td>
<td></td>
</tr>
<tr>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>31m</td>
<td></td>
</tr>
</tbody>
</table>
• Have a copy of the installation program that you used to deploy the cluster.

• Have the files that the installation program generated when you created your cluster.

Procedure

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

```
$ ./openshift-install destroy cluster \n--dir <installation_directory> --log-level info
```

1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. To view different details, specify `warn`, `debug`, or `error` instead of `info`.

```
NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.
```

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.
CHAPTER 5. INSTALLING ON BARE METAL

5.1. INSTALLING A CLUSTER ON BARE METAL

In OpenShift Container Platform version 4.6, you can install a cluster on bare metal infrastructure that you provision.

**IMPORTANT**

While you might be able to follow this procedure to deploy a cluster on virtualized or cloud environments, you must be aware of additional considerations for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you attempt to install an OpenShift Container Platform cluster in such an environment.

5.1.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

5.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

5.1.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.
5.1.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines. If you are running a three-node cluster, running zero compute machines is supported. Running one compute machine is not supported.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

5.1.3.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in **initramfs** during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

5.1.3.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>
1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

5.1.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

5.1.4. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.

2. Provision the required load balancers.

3. Configure the ports for your machines.

4. Configure DNS.

5. Ensure network connectivity.

5.1.4.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config from the machine config server.
During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

**Table 5.1. All machines to all machines**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports <strong>9100-9101</strong> and the Cluster Version Operator on port <strong>9099</strong>.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports <strong>9100-9101</strong>.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

**Table 5.2. All machines to control plane**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

**Table 5.3. Control plane machines to control plane machines**
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 5.4. API load balancer**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>✗</td>
<td>✗</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td></td>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>✗</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
NOTE
The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. Application Ingress load balancer Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

Table 5.5. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

TIP
If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE
A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration
OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources
5.1.4.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 5.6. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
Master hosts <master><n>. <cluster_name>. <base_domain>. DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts <worker><n>. <cluster_name>. <base_domain>. Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

Example 5.1. Sample DNS zone database

```
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5 smtp IN A 192.168.1.5 
helper IN A 192.168.1.5 helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4 IN A 192.168.1.5 api-int.ocp4 IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4 IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4 IN A 192.168.1.97
master1.ocp4 IN A 192.168.1.98
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 5.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root ( 
   2019070700 ; serial 
   3H ; refresh (3 hours) 
   30M ; retry (30 minutes) 
   2W ; expiry (2 weeks) 
   1W ) ; minimum (1 week) 
IN NS ns1.example.com. 
;
; The syntax is "last octet" and the host must have an FQDN 
; with a trailing dot.
97 IN PTR master0.ocp4.example.com. 
98 IN PTR master1.ocp4.example.com. 
99 IN PTR master2.ocp4.example.com. 
;
96 IN PTR bootstrap.ocp4.example.com. 
;
5 IN PTR api.ocp4.example.com. 
5 IN PTR api-int.ocp4.example.com. 
;
11 IN PTR worker0.ocp4.example.com. 
7 IN PTR worker1.ocp4.example.com. 
;
;EOF
```

5.1.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user's ~/.ssh/authorized_keys list.
NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**
   
   Agent pid 31874

   NOTE
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```bash
   $ ssh-add <path>/<file_name> 1
   ```

   **Example output**
   
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

5.1.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**
The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**
Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.1.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

5.1.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:
   
   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:
   
   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

5.1.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:
   
   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
5.1.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

5.1.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:
   
   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

NOTE
You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

IMPORTANT
The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

5.1.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

NOTE
After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT
The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

5.1.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.7. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{.metadata.name}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{ &quot;auths&quot;:{ &quot;cloud.openshift.com&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;you@example.com&quot; }, &quot;quay.io&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;you@example.com&quot; } } }</code></td>
</tr>
</tbody>
</table>

OpenShift Container Platform 4.6 Installing
5.1.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 5.8. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^((32 - 23) - 2)) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
</tbody>
</table>
### Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Table 5.9. Optional parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td>additionalTrustBundle</td>
</tr>
<tr>
<td>compute</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-------------------------</td>
</tr>
<tr>
<td>compute.architecture</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>compute.name</td>
</tr>
<tr>
<td>compute.platform</td>
</tr>
<tr>
<td>compute.replicas</td>
</tr>
<tr>
<td>controlPlane</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>---------------------------------</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><code>controlPlane.hypertreading</code></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><code>controlPlane.name</code></td>
</tr>
<tr>
<td><code>controlPlane.platform</code></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><code>controlPlane.replicas</code></td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
</tr>
<tr>
<td>fips</td>
</tr>
<tr>
<td>imageContentSources</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>---------------------------------</td>
</tr>
<tr>
<td>imageContentSources.source</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
</tr>
<tr>
<td>publish</td>
</tr>
<tr>
<td>sshKey</td>
</tr>
</tbody>
</table>

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

### 5.1.8.2. Sample install-config.yaml file for bare metal

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, `-`, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading (SMT), or **hyperthreading**. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

**NOTE**

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.

**IMPORTANT**

If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml**, ensure that your capacity planning accounts for the dramatically decreased machine performance.

You must set the value of the **replicas** parameter to **0**. This parameter controls the number of
The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines.

The cluster name that you specified in your DNS records.

A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23, then each node is assigned a /23 subnet out of the given `cidr`, which allows for 510 (2^(32 - 23) - 2) pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to `none`. You cannot provide additional platform configuration variables for your platform.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

5.1.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.
NOTE
For bare metal installations, if you do not assign node IP addresses from the range that is specified in the `networking.machineNetwork[].cidr` field in the `install-config.yaml` file, you must include them in the `proxy.noProxy` field.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE
The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
     additionalTrustBundle: | 4
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
     ...
   ```

   - A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   - A proxy URL to use for creating HTTPS connections outside the cluster.
   - A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use * to bypass the proxy for all destinations.
   - If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to
reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

5.1.9. Configuring a three-node cluster

You can optionally install and run three-node clusters in OpenShift Container Platform with no workers. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for development, production, and testing.

Procedure

- Edit the install-config.yaml file to set the number of compute replicas, which are also known as worker replicas, to 0, as shown in the following compute stanza:

```yaml
compute:
  - name: worker
    platform: {}
    replicas: 0
```

5.1.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the install-config.yaml installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   $ ./openshift-install create manifests --dir <installation_directory>  

Example output

   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift

   For `<installation_directory>`, specify the installation directory that contains the install-config.yaml file you created.

**WARNING**

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.
IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become worker nodes.

1. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   c. Save and exit the file.

2. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```
   ├── auth
   │   └── kubeadmin-password
   │           └── kubeconfig
   │        ├── bootstrap.ign
   │        └── master.ign
   │              └── metadata.json
   │                   └── worker.ign
   ```

   Additional resources

   - See [Recovering from expired control plane certificates](#) for more information about recovering kubelet certificates.

5.1.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.
NOTE

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and planned for removal in a future release of OpenShift Container Platform 4.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments:** You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the `APPEND` parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special `coreos.inst.*` arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.

- **Ignition configs:** OpenShift Container Platform Ignition config files (*.ign) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the `coreos-installer` to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.

- **coreos-installer:** You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the `coreos-installer` command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.

NOTE

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

5.1.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.
Procedure

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

```
rhcos-<version>-live.<architecture>.iso
```

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:

- Burn the ISO image to a disk and boot it directly.
- Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.

5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the `coreos-installer`.

6. Run the `coreos-installer` command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

```
$ sudo coreos-installer install \
--ignition-url=https://host/worker.ign /dev/sda
```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.
5.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

Prerequisites

- Obtain the Ignition config files for your cluster.
- Configure suitable PXE or iPXE infrastructure.
- Have access to an HTTP server that you can access from your computer.

Procedure

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**
   
   You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS **kernel**, **initramfs**, and **rootfs** files from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

   The file names contain the OpenShift Container Platform version number. They resemble the following examples:

   - **kernel**: rhcos-<version>-live-kernel-<architecture>
   - **initramfs**: rhcos-<version>-live-initramfs.<architecture>.img
   - **rootfs**: rhcos-<version>-live-rootfs.<architecture>.img

3. Upload the additional files that are required for your booting method:

   - For traditional PXE, upload the **kernel** and **initramfs** files to your TFTP server and the **rootfs** file to your HTTP server.
For iPXE, upload the **kernel**, **initramfs**, and **rootfs** files to your HTTP server.

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images.

   Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

   - For PXE:

     ```
     DEFAULT pxeboot
     TIMEOUT 20
     PROMPT 0
     LABEL pxeboot
     KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture>
     coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
     ```

     **1** Specify the location of the live **kernel** file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

     **2** If you use multiple NICs, specify a single interface in the **ip** option. For example, to use DHCP on a NIC that is named `eno1`, set `ip=eno1:dhcp`.

     **3** Specify locations of the RHCOS files that you uploaded to your HTTP server. The **initrd** parameter value is the location of the **initramfs** file, the **coreos.live.roots_url** parameter value is the location of the **rootfs** file, and the **coreos.inst.ignition_url** parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the **APPEND** line to configure networking or other boot options.

     **NOTE**

     This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the **APPEND** line. For example, add `console=ttty0` `console=ttys0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see *How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?*.

   - For iPXE:

     ```
     kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
     ```
Specify locations of the RHCOS files that you uploaded to your HTTP server. The `kernel` parameter value is the location of the `kernel` file, the `initrd=main` argument is needed for booting on UEFI systems, the `coreos.live.rootfs_url` parameter value is the location of the `rootfs` file, and the `coreos.inst.ignition_url` parameter value is the location of the bootstrap Ignition config file.

If you use multiple NICs, specify a single interface in the `ip` option. For example, to use DHCP on a NIC that is named `eno1`, set `ip=eno1:dhcp`.

Specify the location of the `initramfs` file that you uploaded to your HTTP server.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `kernel` line. For example, add `console=tty0` `console=ttyS0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see *How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?*.

6. If you use PXE UEFI, perform the following actions:

   a. Provide the `shimx64.efi` and `grubx64.efi` EFI binaries and the `grub.cfg` file that are required for booting the system.

      • Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the `images/efiboot.img` file to your host:

      ```
      $ mkdir -p /mnt/iso
      $ mkdir -p /mnt/efiboot
      $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
      $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
      $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
      $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
      $ umount /mnt/efiboot
      ```

      • From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

      ```
      $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
      $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
      ```

      ```
Copy the **EFI/redhat/grub.cfg** file that is included in the RHCOS ISO to your TFTP server.

b. Edit the **grub.cfg** file to include arguments similar to the following:

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
    linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
    <architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
    initrd rhcos-<version>-live-initramfs.<architecture>.img
}
```

where:

* **rhcos-<version>-live-kernel-<architecture>**
  
  Specifies the kernel file that you uploaded to your TFTP server.

  
  Specifies the location of the live rootfs image that you uploaded to your HTTP server.

* **http://<HTTP_server>/bootstrap.ign**
  
  Specifies the location of the bootstrap Ignition config file that you uploaded to your HTTP server.

* **rhcos-<version>-live-initramfs.<architecture>.img**
  
  Specifies the location of the initramfs file that you uploaded to your TFTP server.

**NOTE**

For more information on how to configure a PXE server for UEFI boot, see the Red Hat Knowledgebase article: [How to configure/setup a PXE server for UEFI boot for Red Hat Enterprise Linux?](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Deployment_Guide/s3n101742.html).

7. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

### 5.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
Embedding Ignition configs in an ISO

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

5.1.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

**Procedure**

1. Boot the ISO installer.

2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as `nmcli` or `nmtui`.

3. Run the `coreos-installer` command to install the system, adding the `--copy-network` option to copy networking configuration. For example:

   ```
   $ coreos-installer install --copy-network \ 
      --ignition-url=http://host/worker.ign /dev/sda
   ```

   **IMPORTANT**

   The `--copy-network` option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

5.1.11.3.2. Disk partitioning

The disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless the default partitioning configuration is overridden. During the RHCOS installation, the size of the root file system is increased to use the remaining available space on the target device.
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making `/var` or a subdirectory of `/var`, such as `/var/lib/etcd`, a separate partition, but not both.

  **IMPORTANT**

  Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

### 5.1.11.3.2.1. Creating a separate `/var` partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   ```
3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  # 1
        partitions:
          - label: var
            startMiB: <partition_start_offset>  # 2
            sizeMiB: <partition_size>  # 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
  systemd:
    units:
      - name: var.mount  # 4
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota  # 5
          [Install]
          WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and futurereinstalls of RHCOS might overwrite the beginning of the data partition.
3. The size of the data partition in mebibytes.
4. The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-`

5. The `prjquota` mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
  auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the ISO or PXE manual installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

5.1.1.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the `coreos-installer` command line that causes the installer to maintain one or more existing partitions. For a PXE installation, you can `APPEND coreos.inst.*` options to preserve partitions.

Saved partitions might be partitions from an existing OpenShift Container Platform system that has data partitions that you want to keep. Here are a few tips:

- If you save existing partitions, and those partitions do not leave enough space for RHCOS, installation will fail without damaging the saved partitions.
- Identify the disk partitions you want to keep either by partition label or by number.

**For an ISO installation**

This example preserves any partition in which the partition label begins with `data` (`data*`):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \  
   --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the `coreos-installer` in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \  
   --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \  
   --save-partindex 5- /dev/sda
```
In the previous examples where partition saving is used, `coreos-installer` recreates the partition immediately.

**For a PXE installation**

This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```bash
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```bash
coreos.inst.save_partindex=5-
```

This **APPEND** option preserves partition 6:

```bash
coreos.inst.save_partindex=6
```

5.1.11.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config**: Every manual RHCOS installation needs to pass one of the Ignition config files generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, to carry out the installation.

  **IMPORTANT**
  It is not recommended to modify these files.

  For PXE installations, you pass the Ignition configs on the **APPEND** line using the `coreos.inst.ignition_url=` option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the `coreos-installer` command line with the `--ignition-url=` option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config**: This type must be created manually and should be avoided if possible, as it is not supported by Red Hat. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before and/or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be performed once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

  For PXE or ISO boots, you can create the Ignition config and **APPEND** the `ignition.config.url=` option to identify the location of the Ignition config. You also need to append `ignition.firstboot` `ignition.platform.id=metal` or the `ignition.config.url` option will be ignored.

5.1.11.3.3.1. Embedding an Ignition config in the RHCOS ISO

You can embed a live install Ignition config directly in an RHCOS ISO image. When the ISO image is booted, the embedded config will be applied automatically.

**Procedure**

1. Download the `coreos-installer` binary from the following image mirror page:

   (continued on the next page)
2. Retrieve the RHCOS ISO image and the Ignition config file, and copy them into an accessible directory, such as `/mnt`:

```bash
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. Run the following command to embed the Ignition config into the ISO:

```bash
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \ 
   /mnt/rhcos-<version>-live.x86_64.iso
```

You can now use that ISO to install RHCOS using the specified live install Ignition config.

**IMPORTANT**

Using `coreos-installer iso ignition embed` to embed a file generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, is unsupported and not recommended.

4. To show the contents of the embedded Ignition config and direct it into a file, run:

```bash
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
# diff -s bootstrap.ign mybootstrap.ign
```

**Example output**

Files `bootstrap.ign` and `mybootstrap.ign` are identical

5. To remove the Ignition config and return the ISO to its pristine state so you can reuse it, run:

```bash
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

You can now embed another Ignition config into the ISO or use the ISO in its pristine state.

5.1.11.3.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

**Routing and bonding options at RHCOS boot prompt**

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

**IMPORTANT**

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.
The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

### Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>To configure an IP address, either use DHCP (<code>ip=dhcp</code>) or set an individual static IP address (<code>ip=&lt;host_ip&gt;</code>). Then identify the DNS server IP address (<code>nameserver=&lt;dns_ip&gt;</code>) on each node. This example sets:</td>
<td></td>
</tr>
<tr>
<td>• The node's IP address to <strong>10.10.10.2</strong></td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code></td>
</tr>
<tr>
<td>• The gateway address to <strong>10.10.10.254</strong></td>
<td><code>nameserver=4.4.4.41</code></td>
</tr>
<tr>
<td>• The netmask to <strong>255.255.255.0</strong></td>
<td></td>
</tr>
<tr>
<td>• The hostname to <strong>core0.example.com</strong></td>
<td></td>
</tr>
<tr>
<td>• The DNS server address to <strong>4.4.4.41</strong></td>
<td></td>
</tr>
<tr>
<td>Specify multiple network interfaces by specifying multiple <code>ip=</code> entries.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code> <code>ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Optional: You can configure routes to additional networks by setting an <code>rd.route=</code> value.</td>
<td>To configure the default gateway: <code>ip=::10.10.10.254:::</code></td>
</tr>
<tr>
<td>If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.</td>
<td>To configure the route for the additional network: <code>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</code></td>
</tr>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code> <code>ip=:::core0.example.com:enp2s0:none</code></td>
</tr>
</tbody>
</table>

CHAPTER 5. INSTALLING ON BARE METAL

1033
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can combine DHCP and static IP configurations on systems with multiple network interfaces.</td>
<td>ip=enp1s0:dhcp</td>
</tr>
<tr>
<td></td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>Optional: You can configure VLANs on individual interfaces by using the vlan= parameter.</td>
<td>To configure a VLAN on a network interface and use a static IP address:</td>
</tr>
<tr>
<td></td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0</td>
</tr>
<tr>
<td></td>
<td>To configure a VLAN on a network interface and to use DHCP:</td>
</tr>
<tr>
<td></td>
<td>ip=enp2s0.100:dhcp</td>
</tr>
<tr>
<td></td>
<td>vlan=enp2s0.100:enp2s0</td>
</tr>
<tr>
<td>You can provide multiple DNS servers by adding a nameserver= entry for each server.</td>
<td>nameserver=1.1.1.1</td>
</tr>
<tr>
<td></td>
<td>nameserver=8.8.8.8</td>
</tr>
<tr>
<td>Optional: Bonding multiple network interfaces to a single interface is supported using the bond= option. In these two examples:</td>
<td>To configure the bonded interface to use DHCP, set the bond's IP address to dhcp. For example:</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup</td>
</tr>
<tr>
<td></td>
<td>ip=bond0:dhcp</td>
</tr>
<tr>
<td></td>
<td>To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup</td>
</tr>
<tr>
<td></td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none</td>
</tr>
</tbody>
</table>

- The syntax for configuring a bonded interface is: `bond=name[:network_interfaces] [:options]`  
- `name` is the bonding device name (bond0), `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (em1, em2), and `options` is a comma-separated list of bonding options. Enter `modinfo bonding` to see available options.  
- When you create a bonded interface using bond=, you must specify how the IP address is assigned and other information for the bonded interface.
### Description
Optional: You can configure VLANs on bonded interfaces by using the `vlan=` parameter.

### Examples
To configure the bonded interface with a VLAN and to use DHCP:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

To configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

### Description
Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example:

- The syntax for configuring a team interface is: `team=name[:network_interfaces]`
  - `name` is the team device name (team0) and `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (`em1, em2`).

### Examples
To configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

---

**coreos.inst boot options for ISO or PXE install**

While you can pass most standard installation boot arguments to the live installer, there are several arguments that are specific to the RHCOS live installer.

- For ISO, these options can be added by interrupting the RHCOS installer.
- For PXE or iPXE, these options must be added to the `APPEND` line before starting the PXE kernel. You cannot interrupt a live PXE install.

The following table shows the RHCOS live installer boot options for ISO and PXE installs.

**Table 5.10. coreos.inst boot options**

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| Example 1: Booting with VLAN                     | `ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0`                                                        |
| Example 2: Booting with Static IP                | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0`                                                        |
| Example 3: Booting with VLAN and Network Team    | `team=team0:em1,em2
ip=team0:dhcp`                                                              |
<p>| <strong>NOTE</strong>                                          | Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this Red Hat Knowledgebase Article. |</p>
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>coreos.inst.install_dev</code></td>
<td>Required. The block device on the system to install to. It is recommended to use the full path, such as <code>/dev/sda</code>, although <code>sda</code> is allowed.</td>
</tr>
<tr>
<td><code>coreos.inst.ignition_url</code></td>
<td>Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded.</td>
</tr>
<tr>
<td><code>coreos.inst.save_partlabel</code></td>
<td>Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td><code>coreos.inst.save_partindex</code></td>
<td>Optional: Comma-separated indexes of partitions to preserve during the install. Ranges <code>m-n</code> are permitted, and either <code>m</code> or <code>n</code> can be omitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td><code>coreos.inst.insecure</code></td>
<td>Optional: Permits the OS image that is specified by <code>coreos.inst.image_url</code> to be unsigned.</td>
</tr>
<tr>
<td><code>coreos.inst.image_url</code></td>
<td>Optional: Download and install the specified RHCOS image.</td>
</tr>
<tr>
<td></td>
<td>• This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
<tr>
<td></td>
<td>• While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install.</td>
</tr>
<tr>
<td></td>
<td>• If you are using <code>coreos.inst.image_url</code>, you must also use <code>coreos.inst.insecure</code>. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform.</td>
</tr>
<tr>
<td></td>
<td>• Only HTTP and HTTPS protocols are supported.</td>
</tr>
<tr>
<td><code>coreos.inst.skip_reboot</code></td>
<td>Optional: The system will not reboot after installing. Once the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
</tbody>
</table>
coreos.inst.platform_id

Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal. This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: `coreos.inst.platform_id=vmware`.

ignition.config.url

Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from `coreos.inst.ignition_url`, which is the Ignition config for the installed system.

**coreos-installer options for ISO install**

You can also install RHCOS by invoking the coreos-installer command directly from the command line. The kernel arguments in the previous table provide a shortcut for automatically invoking coreos-installer at boot time, but you can pass similar arguments directly to coreos-installer when running it from a shell prompt.

The following table shows the options and subcommands you can pass to the coreos-installer command from a shell prompt during a live install.

**Table 5.11. coreos-installer command-line options, arguments, and subcommands**

<table>
<thead>
<tr>
<th>Command-line options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-u, --image-url &lt;url&gt;</td>
<td>Specify the image URL manually.</td>
</tr>
<tr>
<td>-f, --image-file &lt;path&gt;</td>
<td>Specify a local image file manually.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>Embed an Ignition config from a file.</td>
</tr>
<tr>
<td>-I, --ignition-url &lt;URL&gt;</td>
<td>Embed an Ignition config from a URL.</td>
</tr>
<tr>
<td>--ignition-hash &lt;digest&gt;</td>
<td>Digest type-value of the Ignition config.</td>
</tr>
<tr>
<td>-p, --platform &lt;name&gt;</td>
<td>Override the Ignition platform ID.</td>
</tr>
<tr>
<td>--append-karg &lt;arg&gt;...</td>
<td>Append the default kernel argument.</td>
</tr>
<tr>
<td>--delete-karg &lt;arg&gt;...</td>
<td>Delete the default kernel argument.</td>
</tr>
</tbody>
</table>
**-n, --copy-network**

Copy the network configuration from the install environment.

**IMPORTANT**

The `--copy-network` option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>--network-dir &lt;path&gt;</strong></td>
<td>For use with <code>-n</code>. Default is <code>/etc/NetworkManager/system-connections/</code>.</td>
</tr>
<tr>
<td><strong>--save-partlabel &lt;lx&gt;..</strong></td>
<td>Save partitions with this label glob.</td>
</tr>
<tr>
<td><strong>--save-partindex &lt;id&gt;...</strong></td>
<td>Save partitions with this number or range.</td>
</tr>
<tr>
<td><strong>--offline</strong></td>
<td>Force offline installation.</td>
</tr>
<tr>
<td><strong>--insecure</strong></td>
<td>Skip signature verification.</td>
</tr>
<tr>
<td><strong>--insecure-ignition</strong></td>
<td>Allow Ignition URL without HTTPS or hash.</td>
</tr>
<tr>
<td><strong>--architecture &lt;name&gt;</strong></td>
<td>Target CPU architecture. Default is <code>x86_64</code>.</td>
</tr>
<tr>
<td><strong>--preserve-on-error</strong></td>
<td>Do not clear partition table on error.</td>
</tr>
<tr>
<td><strong>-h, --help</strong></td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

**Command-line argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;device&gt;</code></td>
<td>The destination device.</td>
</tr>
</tbody>
</table>

**coreos-installer embedded Ignition commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ coreos-installer iso ignition embed &lt;options&gt; --ignition-file &lt;file_path&gt; &lt;ISO_image&gt;</code></td>
<td>Embed an Ignition config in an ISO image.</td>
</tr>
<tr>
<td><code>coreos-installer iso ignition show &lt;options&gt; &lt;ISO_image&gt;</code></td>
<td>Show the embedded Ignition config from an ISO image.</td>
</tr>
</tbody>
</table>
coreos-installer iso ignition remove <options> <ISO_image>
Remove the embedded Ignition config from an ISO image.

coreos-installer ISO Ignition options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f, --force</td>
<td>Overwrite an existing Ignition config.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

coreos-installer PXE Ignition commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreos-installer pxe ignition wrap &lt;options&gt;</td>
<td>Wrap an Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;image_name&gt;</td>
<td>Show the wrapped Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;initrd_name&gt;</td>
<td>Show the wrapped Ignition config in an initrd image.</td>
</tr>
</tbody>
</table>

coreos-installer PXE Ignition options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

5.1.12. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
You obtained the installation program and generated the Ignition config files for your cluster.

You used the Ignition config files to create RHCOS machines for your cluster.

Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**

1. Monitor the bootstrap process:

   ```sh
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **Example output**

   ```
   INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
   INFO API v1.19.0 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   INFO It is now safe to remove the bootstrap resources
   ```

   The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   **IMPORTANT**

   You must remove the bootstrap machine from the load balancer at this point.
   You can also remove or reformat the machine itself.

5.1.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.

- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```sh
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```
1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:
   ```bash
   $ oc whoami
   
   Example output
   
   $ system:admin
   
5.1.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:
   ```bash
   $ oc get nodes
   
   Example output
   
   NAME      STATUS    ROLES   AGE     VERSION
   master-0  Ready     master  63m     v1.19.0
   master-1  Ready     master  63m     v1.19.0
   master-2  Ready     master  64m     v1.19.0
   
   The output lists all of the machines that you created.
   
   NOTE
   
   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.
   
2. Review the pending CSRs and ensure that you see the client requests with the Pending or Approved status for each machine that you added to the cluster:
   ```bash
   $ oc get csr
   
   Example output
   
   NAME     AGE     REQUESTOR                                                                 CONDITION
   csr-8b2br 15m     system:serviceaccount:openshift-machine-config-operator:node-
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

   **NOTE**
   
   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

   **NOTE**
   
   For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

   - To approve them individually, run the following command for each valid CSR:

     $ oc adm certificate approve <csr_name>

     **<csr_name>** is the name of a CSR from the list of current CSRs.

   - To approve all pending CSRs, run the following command:

     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve

     **NOTE**
     
     Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:
5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name>
  ```

  - `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

   ```bash
   $ oc get nodes
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

   **NOTE**

   It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

**5.1.15. Initial Operator configuration**
After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

   ```bash
   $ watch -n5 oc get clusteroperators
   
   Example output
   ```

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

**5.1.15.1. Image registry removed during installation**
On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE
The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

5.1.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

5.1.15.2.1. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

IMPORTANT

OpenShift Container Platform supports ReadWriteOnce access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, ReadWriteMany access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the spec.storage.pvc in the configs.imageregistry/cluster resource.
NOTE

When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   $ oc get pod -n openshift-image-registry

NOTE

If the storage type is emptyDIR, the replica number cannot be greater than 1.

3. Check the registry configuration:

   $ oc edit config.imageregistry.operator.openshift.io

   Example output

   ```yaml
   storage:
   pvc:
   claim:
   ```

   Leave the claim field blank to allow the automatic creation of an  image-registry-storage PVC.

4. Check the clusteroperator status:

   $ oc get clusteroperator image-registry

5. Ensure that your registry is set to managed to enable building and pushing of images.

   • Run:

     $ oc edit config.imageregistry/cluster

     Then, change the line

     ```yaml
     managementState: Removed
     ```

     to

     ```yaml
     managementState: Managed
     ```

---

5.15.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

• To set the image registry storage to an empty directory:
WARNING
Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

5.1.15.2.3. Configuring block registry storage

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**
Block storage volumes are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only one (1) replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

3. Edit the registry configuration so that it references the correct PVC.

5.1.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.
Procedure

1. Confirm that all the cluster components are online with the following command:

   ```
   $ watch -n5 oc get clusteroperators
   ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td></td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```
The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```bash
   $ oc get pods --all-namespaces
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2m</td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```bash
   $ oc logs <pod_name> -n <namespace>
   ```

   **Specify the pod name and namespace, as shown in the output of the previous command.**

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.
5.1.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

5.1.18. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

5.2. INSTALLING A CLUSTER ON BARE METAL WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on bare metal infrastructure that you provision with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only kubeProxy configuration parameters in a running cluster.

5.2.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to access Red Hat Insights.

5.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

5.2.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

5.2.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines. If you are running a three-node cluster, running zero compute machines is supported. Running one compute machine is not supported.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

5.2.3.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.
5.2.3.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

5.2.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

5.2.4. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

### 5.2.4.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

#### Table 5.12. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

#### Table 5.13. All machines to control plane
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 5.15. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
</tbody>
</table>
Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer** Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<p>| Table 5.16. Application Ingress load balancer |
|-----------------------------------------------|-----------------------------------------------|</p>
<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.
NTP configuration

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- Configuring chrony time service

5.2.4.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 5.17. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
### Component | Record | Description
--- | --- | ---
Routes | *.apps.<cluster_name>.<base_domain> | Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Bootstrap | bootstrap.<cluster_name>.<base_domain> | Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

Master hosts | <master><n>.<cluster_name>.<base_domain> | DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts | <worker><n>.<cluster_name>.<base_domain> | Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

### Example 5.3. Sample DNS zone database

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
The api identifies the IP of your load balancer.
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 5.4. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (  
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
  ; The syntax is "last octet" and the host must have an FQDN
  ; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
  ;
96 IN PTR bootstrap.ocp4.example.com.
  ;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
  ;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
  ;
;EOF
```

5.2.5. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an
SSH key to both your `ssh-agent` and the installation program. You can use this key to access the
bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the
key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches
such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your
   computer, create one. For example, on a computer that uses a Linux operating system, run the
   following command:

   ```bash
   $ ssh-keygen -t ed25519 -N " 
   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an
   existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location
   that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS
   Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead,
   create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate
   the SSH key. The key must be either RSA or ECDSA.
3. Add your SSH private key to the **ssh-agent**:

   ```
   $ ssh-add <path>/<file_name>
   ```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1. Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 5.2.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

5.2.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

5.2.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

5.2.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

```
C:\> path
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 5.2.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 5.2.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   NOTE
   
   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   IMPORTANT
   
   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

5.2.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

   NOTE
   
   After installation, you cannot modify these parameters in the `install-config.yaml` file.

   IMPORTANT
   
   The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

5.2.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>

Table 5.18. Required parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;, &lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 5.2.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 5.19. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>networking</code></td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong> You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
<td></td>
</tr>
<tr>
<td><code>networking.network</code></td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td><code>networking.clusterNetwork</code></td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>10.128.0.0/14</code> with a host prefix of <code>/23</code>.</td>
<td>networking:</td>
</tr>
</tbody>
</table>
|                    | If you specify multiple IP address blocks, the blocks must not overlap. | clusterNetwork:
<p>|                    | - cidr: <code>10.128.0.0/14</code>                                          | - hostPrefix: <code>23</code> |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking: serviceNetwork: - 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking: machineNetwork: - cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block.</td>
<td>An IP network block in CIDR notation.</td>
</tr>
<tr>
<td></td>
<td>An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
</tr>
</tbody>
</table>

5.2.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 5.20. Optional parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hypervthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong> If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (“”).</td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**
The use of FIPS Validated Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**
If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <code>false</code> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms. <strong>IMPORTANT</strong> If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to <a href="#">BZ#1953035</a>.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines. <strong>NOTE</strong> For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.</td>
<td>One or more keys. For example: sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</td>
</tr>
</tbody>
</table>

### 5.2.8.2. Sample install-config.yaml file for bare metal

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute: 2
- hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane: 5
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: test
```
networking:
- cidr: 10.128.0.0/14
- hostPrefix: 23
networkType: OpenShiftSDN
serviceNetwork: 172.30.0.0/16
platform:
- none: {}
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAAA...'

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2. The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not.
   Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3. Whether to enable or disable simultaneous multithreading (SMT), or **hyperthreading**. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

   **NOTE**
   
   Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.

   **IMPORTANT**
   
   If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml**, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4. You must set the value of the **replicas** parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

5. The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

6. The cluster name that you specified in your DNS records.

7. A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.
The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23, then each node is assigned a /23 subnet out of the given `cidr`, which allows for 510 ($2^{32 - 23} - 2$)

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to `none`. You cannot provide additional platform configuration variables for your platform.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 5.2.9. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".
NOTE

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

Phase 2

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

5.2.10. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- Create the `install-config.yaml` file and complete any modifications to it.
- Create the Ignition config files for your cluster.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`

   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```bash
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```
where:

<installation_directory>

Specifies the directory name that contains the manifests/ directory for your cluster.

3. Open the cluster-network-03-config.yml file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

**Specify a different VXLAN port for the OpenShift SDN network provider**

```yaml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
defaultNetwork:
  openshiftSDNConfig:
    vxlanPort: 4800
```

4. Save the cluster-network-03-config.yml file and quit the text editor.

5. Optional: Back up the manifests/cluster-network-03-config.yml file. The installation program deletes the manifests/ directory when creating the cluster.

### 5.2.11. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named cluster. The CR specifies the fields for the Network API in the operator.openshift.io API group.

The CNO configuration inherits the following fields during cluster installation from the Network API in the Network.config.openshift.io API group and these fields cannot be changed:

- **clusterNetwork**
  - IP address pools from which pod IP addresses are allocated.
- **serviceNetwork**
  - IP address pool for services.
- **defaultNetwork.type**
  - Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the defaultNetwork object in the CNO object named cluster.

### 5.2.11.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always cluster.</td>
</tr>
</tbody>
</table>
spec.clusterNetwork array

A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:

```
spec:
  clusterNetwork:
    - cidr: 10.128.0.0/19
      hostPrefix: 23
    - cidr: 10.128.32.0/19
      hostPrefix: 23
```

This value is ready-only and specified in the `install-config.yaml` file.

spec.serviceNetwork array

A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:

```
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.

spec.defaultNetwork object

Configures the Container Network Interface (CNI) cluster network provider for the cluster network.

spec.kubeProxy Config object

The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.

defaultNetwork object configuration
The values for the `defaultNetwork` object are defined in the following table:

---

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultNetwork</td>
<td>object</td>
<td>Configures the Container Network Interface (CNI) cluster network provider for the cluster network.</td>
</tr>
<tr>
<td>kubeProxy</td>
<td>object</td>
<td>The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.</td>
</tr>
</tbody>
</table>
Either OpenShiftSDN or OVNKubernetes. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.

**NOTE**

OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>Either OpenShiftSDN or OVNKubernetes. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshiftSDNConfig</td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td>ovnKubernetesConfig</td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

Configuration for the OpenShift SDN CNI cluster network provider

The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

Table 5.23. openshiftSDNConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</td>
</tr>
</tbody>
</table>

The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.
The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.

This value cannot be changed after cluster installation.

The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.

If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.

On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.

### Example OpenShift SDN configuration

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

### Configuration for the OVN-Kubernetes CNI cluster network provider

The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>vxlanPort</td>
<td>integer</td>
<td>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation. If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number. On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</td>
</tr>
</tbody>
</table>

### Table 5.24. ovnKubernetesConfig object
### Field Types and Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>genevePort</td>
<td>integer</td>
<td>The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

### Example OVN-Kubernetes Configuration

```yaml
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

### kubeProxyConfig object configuration

The values for the `kubeProxyConfig` object are defined in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultNetwork</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ovnKubernetesConfig</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtu</td>
<td></td>
<td></td>
</tr>
<tr>
<td>genevePort</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for <code>iptables</code> rules. The default value is 30s. Valid suffixes include <code>s</code>, <code>m</code>, and <code>h</code> and are described in the Go time package documentation.</td>
</tr>
</tbody>
</table>

**NOTE**

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the `iptablesSyncPeriod` parameter is no longer necessary.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>proxyArguments.iptables-min-sync-period</td>
<td>array</td>
<td>The minimum duration before refreshing <code>iptables</code> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include <code>s</code>, <code>m</code>, and <code>h</code> and are described in the Go time package. The default value is:</td>
</tr>
</tbody>
</table>

```
kubeProxyConfig:
  proxyArguments:
    iptables-min-sync-period:
    - 0s
```

### 5.2.12. Creating the Ignition config files

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.
Procedure

- Obtain the Ignition config files:

  $ ./openshift-install create ignition-configs --dir <installation_directory>

  For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

  **IMPORTANT**

  If you created an `install-config.yaml` file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubectlconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5.2.13. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.

**NOTE**

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and planned for removal in a future release of OpenShift Container Platform 4.

You can configure RHCOS during ISO and PXE installations by using the following methods:
Kernel arguments: You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the APPEND parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special coreos.inst.* arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.

Ignition configs: OpenShift Container Platform Ignition config files (*.ign) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the coreos-installer to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.

coreos-installer: You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the coreos-installer command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.

NOTE

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

5.2.13.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

Prerequisites

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.

Procedure

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

IMPORTANT

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.
2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

```
rhcos-<version>-live.<architecture>.iso
```

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:

- Burn the ISO image to a disk and boot it directly.
- Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.

5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the `coreos-installer`.

6. Run the `coreos-installer` command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

```
$ sudo coreos-installer install \
   --ignition-url=https://host/worker.ign /dev/sda
```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

### 5.2.13.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

**Prerequisites**
Obtain the Ignition config files for your cluster.

Configure suitable PXE or iPXE infrastructure.

Have access to an HTTP server that you can access from your computer.

**Procedure**

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**
   
   You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS **kernel**, **initramfs** and **rootfs** files from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

   The file names contain the OpenShift Container Platform version number. They resemble the following examples:
   
   - **kernel**: `rhcos-<version>-live-kernel-<architecture>`
   - **initramfs**: `rhcos-<version>-live-initramfs.<architecture>.img`
   - **rootfs**: `rhcos-<version>-live-rootfs.<architecture>.img`

3. Upload the additional files that are required for your booting method:

   - For traditional PXE, upload the **kernel** and **initramfs** files to your TFTP server and the **rootfs** file to your HTTP server.
   
   - For iPXE, upload the **kernel**, **initramfs**, and **rootfs** files to your HTTP server.

   **IMPORTANT**
   
   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images.
Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE:

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture>
        coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
```

1. Specify the location of the live kernel file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

2. If you use multiple NICs, specify a single interface in the `ip` option. For example, to use DHCP on a NIC that is named `eno1`, set `ip=eno1:dhcp`.

3. Specify locations of the RHCOS files that you uploaded to your HTTP server. The `initrd` parameter value is the location of the `initramfs` file, the `coreos.live.rootsfs_url` parameter value is the location of the `rootsfs` file, and the `coreos.inst.ignition_url` parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the `APPEND` line to configure networking or other boot options.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `APPEND` line. For example, add `console=tty0 console=ttyS0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see *How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?*

- For iPXE:

```
kernal http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
        <architecture>.img coreos.inst.install_dev=/dev/sda
        coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
        <architecture>.img
```

1. Specify locations of the RHCOS files that you uploaded to your HTTP server. The `kernel` parameter value is the location of the `kernel` file, the `initrd=main` argument is needed for booting on UEFI systems, the `coreos.live.rootsfs_url` parameter value is the location of the `rootsfs` file, and the `coreos.inst.ignition_url` parameter value is the location of the bootstrap Ignition config file.
If you use multiple NICs, specify a single interface in the `ip` option. For example, to use DHCP on a NIC that is named `eno1`, set `ip=eno1:dhcp`.

Specify the location of the `initramfs` file that you uploaded to your HTTP server.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `kernel` line. For example, add `console=tty0 console=ttyS0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?

6. If you use PXE UEFI, perform the following actions:

   a. Provide the `shimx64.efi` and `grubx64.efi` EFI binaries and the `grub.cfg` file that are required for booting the system.

      - Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the `images/efiboot.img` file to your host:

        ```
        $ mkdir -p /mnt/iso
        $ mkdir -p /mnt/efiboot
        $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
        $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
        $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
        $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
        $ umount /mnt/efiboot
        $ umount /mnt/iso
        ```

      - From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

        ```
        $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
        $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
        $ umount /mnt/efiboot
        $ umount /mnt/iso
        ```

      - Copy the `EFI/redhat/grub.cfg` file that is included in the RHCOS ISO to your TFTP server.

   b. Edit the `grub.cfg` file to include arguments similar to the following:

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
        linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
      }
      ```
where:

rhcos-<version>-live-kernel-<architecture>  
Specifies the **kernel** file that you uploaded to your TFTP server.

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img  
Specifies the location of the live rootfs image that you uploaded to your HTTP server.

http://<HTTP_server>/bootstrap.ign  
Specifies the location of the bootstrap Ignition config file that you uploaded to your HTTP server.

rhcos-<version>-live-initramfs.<architecture>.img  
Specifies the location of the **initramfs** file that you uploaded to your TFTP server.

**NOTE**

For more information on how to configure a PXE server for UEFI boot, see the Red Hat Knowledgebase article: How to configure/setup a PXE server for UEFI boot for Red Hat Enterprise Linux?.

7. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

5.2.13.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
- Embedding Ignition configs in an ISO

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

5.2.13.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:
• Pass special kernel parameters when you boot the live installer.

• Use a machine config to copy networking files to the installed system.

• Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

• See the "Advanced RHCOS installation reference" tables.

• Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

Procedure

1. Boot the ISO installer.

2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as nmcli or nmtui.

3. Run the coreos-installer command to install the system, adding the --copy-network option to copy networking configuration. For example:

   ```
   $ coreos-installer install --copy-network \n   --ignition-url=http://host/worker.ign /dev/sda
   ```

   **IMPORTANT**
   
   The --copy-network option only copies networking configuration found under /etc/NetworkManager/system-connections. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

5.2.13.3.2. Disk partitioning

The disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless the default partitioning configuration is overridden. During the RHCOS installation, the size of the root file system is increased to use the remaining available space on the target device.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

• Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making /var or a subdirectory of /var, such as /var/lib/etcd, a separate partition, but not both.

   **IMPORTANT**
   
   Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.
• Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

5.2.13.3.2.1. Creating a separate `/var` partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:
   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:
   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadmin-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...
   ```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:
   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   ```
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
  storage:
    disks:
      - device: /dev/<device_name> ①
    partitions:
      - label: var
        startMiB: <partition_start_offset> ②
        sizeMiB: <partition_size> ③
    filesystems:
      - device: /dev/disk/by-partlabel/var
        path: /var
        format: xfs
  systemd:
    units:
      - name: var.mount ④
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota ⑤
          [Install]
          WantedBy=local-fs.target

① The storage device name of the disk that you want to partition.
② When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
③ The size of the data partition in mebibytes.
④ The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.
⑤ The prjquota mount option must be enabled for filesystems used for container storage.
NOTE

When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
  auth bootstrap.ign master.ign metadata.json worker.ign

Now you can use the Ignition config files as input to the ISO or PXE manual installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

5.2.13.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the coreos-installer command line that causes the installer to maintain one or more existing partitions. For a PXE installation, you can APPEND coreos.inst.* options to preserve partitions.

Saved partitions might be partitions from an existing OpenShift Container Platform system that has data partitions that you want to keep. Here are a few tips:

- If you save existing partitions, and those partitions do not leave enough space for RHCOS, installation will fail without damaging the saved partitions.
- Identify the disk partitions you want to keep either by partition label or by number.

For an ISO installation

This example preserves any partition in which the partition label begins with data (data*):

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign 
  --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the coreos-installer in a way that preserves the sixth (6) partition on the disk:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign 
  --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign 
  --save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, coreos-installer recreates the partition immediately.

For a PXE installation

This APPEND option preserves any partition in which the partition label begins with 'data' ('data*'):
This APPEND option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

This APPEND option preserves partition 6:

```
coreos.inst.save_partindex=6
```

5.2.13.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config**: Every manual RHCOS installation needs to pass one of the Ignition config files generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, to carry out the installation.

  IMPORTANT

  It is not recommended to modify these files.

  For PXE installations, you pass the Ignition configs on the APPEND line using the `coreos.inst.ignition_url=` option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the `coreos-installer` command line with the `--ignition-url=` option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config**: This type must be created manually and should be avoided if possible, as it is not supported by Red Hat. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before and/or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be performed once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

  For PXE or ISO boots, you can create the Ignition config and APPEND the `ignition.config.url=` option to identify the location of the Ignition config. You also need to append `ignition.firstboot` `ignition.platform.id=metal` or the `ignition.config.url` option will be ignored.

5.2.13.3.3.1. Embedding an Ignition config in the RHCOS ISO

You can embed a live install Ignition config directly in an RHCOS ISO image. When the ISO image is booted, the embedded config will be applied automatically.

**Procedure**

1. Download the `coreos-installer` binary from the following image mirror page:

2. Retrieve the RHCOS ISO image and the Ignition config file, and copy them into an accessible directory, such as `/mnt`:

   ```
   # cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
   # chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
   ```
3. Run the following command to embed the Ignition config into the ISO:

```bash
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign
  /mnt/rhcos-<version>-live.x86_64.iso
```

You can now use that ISO to install RHCOS using the specified live install Ignition config.

**IMPORTANT**

Using `coreos-installer iso ignition embed` to embed a file generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, is unsupported and not recommended.

4. To show the contents of the embedded Ignition config and direct it into a file, run:

```bash
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
# diff -s bootstrap.ign mybootstrap.ign
```

**Example output**

Files `bootstrap.ign` and `mybootstrap.ign` are identical

5. To remove the Ignition config and return the ISO to its pristine state so you can reuse it, run:

```bash
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

You can now embed another Ignition config into the ISO or use the ISO in its pristine state.

### 5.2.13.3.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

**Routing and bonding options at RHCOS boot prompt**

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

**IMPORTANT**

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.
**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

### Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the *dracut* tool during system boot. For more information about the networking options supported by *dracut*, see the *dracut.cmdline* manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| To configure an IP address, either use DHCP (`ip=dhcp`) or set an individual static IP address (`ip=<host_ip>`). Then identify the DNS server IP address (`nameserver=<dns_ip>`) on each node. This example sets: | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41` |
| - The node’s IP address to **10.10.10.2**                                   |                                                                          |
| - The gateway address to **10.10.10.254**                                   |                                                                          |
| - The netmask to **255.255.255.0**                                         |                                                                          |
| - The hostname to **core0.example.com**                                     |                                                                          |
| - The DNS server address to **4.4.4.41**                                    |                                                                          |
| Specify multiple network interfaces by specifying multiple `ip=` entries.    | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none` |
| Optional: You can configure routes to additional networks by setting an `rd.route=` value. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway. | To configure the default gateway:                                        |
|                                                                           | `ip=::10.10.10.254:::`                                                  |
|                                                                           | To configure the route for the additional network:                      |
|                                                                           | `rd.route=20.20.20.0/24:20.20.20.254:enp2s0`                           |
| Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=:::core0.example.com:enp2s0:none` |
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can combine DHCP and static IP configurations on systems with multiple network interfaces.</td>
<td>ip=enp1s0:dhcp&lt;br&gt;ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>Optional: You can configure VLANs on individual interfaces by using the <code>vlan=</code> parameter.</td>
<td>To configure a VLAN on a network interface and use a static IP address:&lt;br&gt;ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none&lt;br&gt;vlan=enp2s0.100:enp2s0&lt;br&gt;To configure a VLAN on a network interface and to use DHCP:&lt;br&gt;ip=enp2s0.100:dhcp&lt;br&gt;vlan=enp2s0.100:enp2s0</td>
</tr>
<tr>
<td>You can provide multiple DNS servers by adding a <code>nameserver=</code> entry for each server.</td>
<td>nameserver=1.1.1.1&lt;br&gt;nameserver=8.8.8.8</td>
</tr>
<tr>
<td>Optional: Bonding multiple network interfaces to a single interface is supported using the <code>bond=</code> option. In these two examples:</td>
<td>To configure the bonded interface to use DHCP, set the bond’s IP address to <code>dhcp</code>. For example:&lt;br&gt;bond=bond0:em1,em2:mode=active-backup&lt;br&gt;ip=bond0:dhcp&lt;br&gt;To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:&lt;br&gt;bond=bond0:em1,em2:mode=active-backup&lt;br&gt;ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none</td>
</tr>
<tr>
<td>- The syntax for configuring a bonded interface is: <code>bond=name[:network_interfaces][:options]</code>&lt;br&gt;- <code>name</code> is the bonding device name (<code>bond0</code>), <code>network_interfaces</code> represents a comma-separated list of physical (ethernet) interfaces (<code>em1,em2</code>), and <code>options</code> is a comma-separated list of bonding options. Enter <code>modinfo bonding</code> to see available options.&lt;br&gt;- When you create a bonded interface using <code>bond=</code>, you must specify how the IP address is assigned and other information for the bonded interface.</td>
<td></td>
</tr>
</tbody>
</table>
Optional: You can configure VLANs on bonded interfaces by using the `vlan=` parameter.

To configure the bonded interface with a VLAN and to use DHCP:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

To configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example:

```
team=team0:em1,em2
ip=team0:dhcp
```

**NOTE**

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](https://access.redhat.com/articles/).

---

**coreos.inst boot options for ISO or PXE install**

While you can pass most standard installation boot arguments to the live installer, there are several arguments that are specific to the RHCOS live installer.

- For ISO, these options can be added by interrupting the RHCOS installer.

- For PXE or iPXE, these options must be added to the `APPEND` line before starting the PXE kernel. You cannot interrupt a live PXE install.

The following table shows the RHCOS live installer boot options for ISO and PXE installs.

**Table 5.26. coreos.inst boot options**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreos.inst</td>
<td>boot options for ISO or PXE install</td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>coreos.inst.install_dev</td>
<td>Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda, although sda is allowed.</td>
</tr>
<tr>
<td>coreos.inst.ignition_url</td>
<td>Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded.</td>
</tr>
<tr>
<td>coreos.inst.save_partlabel</td>
<td>Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.save_partindex</td>
<td>Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.insecure</td>
<td>Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.</td>
</tr>
<tr>
<td>coreos.inst.image_url</td>
<td>Optional: Download and install the specified RHCOS image.</td>
</tr>
<tr>
<td></td>
<td>- This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
<tr>
<td></td>
<td>- While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install.</td>
</tr>
<tr>
<td></td>
<td>- If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform.</td>
</tr>
<tr>
<td></td>
<td>- Only HTTP and HTTPS protocols are supported.</td>
</tr>
<tr>
<td>coreos.inst.skip_reboot</td>
<td>Optional: The system will not reboot after installing. Once the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
</tbody>
</table>
coreos.inst.platform_id

Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal. This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example: coreos.inst.platform_id=vmware.

ignition.config.url

Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how coreos-installer is invoked, or to run code before or after the installation. This is different from coreos.inst.ignition_url, which is the Ignition config for the installed system.

coreos-installer options for ISO install

You can also install RHCOS by invoking the coreos-installer command directly from the command line. The kernel arguments in the previous table provide a shortcut for automatically invoking coreos-installer at boot time, but you can pass similar arguments directly to coreos-installer when running it from a shell prompt.

The following table shows the options and subcommands you can pass to the coreos-installer command from a shell prompt during a live install.

Table 5.27. coreos-installer command-line options, arguments, and subcommands

<table>
<thead>
<tr>
<th>Command-line options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>-u, --image-url &lt;url&gt;</td>
<td>Specify the image URL manually.</td>
</tr>
<tr>
<td>-f, --image-file &lt;path&gt;</td>
<td>Specify a local image file manually.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>Embed an Ignition config from a file.</td>
</tr>
<tr>
<td>-I, --ignition-url &lt;URL&gt;</td>
<td>Embed an Ignition config from a URL.</td>
</tr>
<tr>
<td>--ignition-hash &lt;digest&gt;</td>
<td>Digest type-value of the Ignition config.</td>
</tr>
<tr>
<td>-p, --platform &lt;name&gt;</td>
<td>Override the Ignition platform ID.</td>
</tr>
<tr>
<td>--append-karg &lt;arg&gt;...</td>
<td>Append the default kernel argument.</td>
</tr>
<tr>
<td>--delete-karg &lt;arg&gt;...</td>
<td>Delete the default kernel argument.</td>
</tr>
</tbody>
</table>
-n, --copy-network  Copy the network configuration from the install environment.

**IMPORTANT**

The **--copy-network** option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

--network-dir <path>  For use with -n. Default is `/etc/NetworkManager/system-connections/`.

--save-partlabel <lx>..  Save partitions with this label glob.

--save-partindex <id>...  Save partitions with this number or range.

--offline  Force offline installation.

--insecure  Skip signature verification.

--insecure-ignition  Allow Ignition URL without HTTPS or hash.

--architecture <name>  Target CPU architecture. Default is x86_64.

--preserve-on-error  Do not clear partition table on error.

-h, --help  Print help information.

**Command-line argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;device&gt;</td>
<td>The destination device.</td>
</tr>
</tbody>
</table>

**coreos-installer embedded Ignition commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ coreos-installer iso ignition embed &lt;options&gt; --ignition-file &lt;file_path&gt; &lt;ISO_image&gt;</td>
<td>Embed an Ignition config in an ISO image.</td>
</tr>
<tr>
<td>coreos-installer iso ignition show &lt;options&gt; &lt;ISO_image&gt;</td>
<td>Show the embedded Ignition config from an ISO image.</td>
</tr>
</tbody>
</table>
coreos-installer iso ignition remove
<options> <ISO_image>

Remove the embedded Ignition config from an ISO image.

coreos-installer ISO Ignition options

Option | Description
--- | ---
-f, --force | Overwrite an existing Ignition config.
-i, --ignition-file <path> | The Ignition config to be used. Default is stdin.
-o, --output <path> | Write the ISO to a new output file.
-h, --help | Print help information.

coreos-installer PXE Ignition commands

Command | Description
--- | ---

Note that not all of these options are accepted by all subcommands.

coreos-installer pxe ignition wrap <options> | Wrap an Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <image_name> | Show the wrapped Ignition config in an image.
coreos-installer pxe ignition unwrap <options> <initrd_name> | Show the wrapped Ignition config in an initrd image.

coreos-installer PXE Ignition options

Option | Description
--- | ---
-i, --ignition-file <path> | The Ignition config to be used. Default is stdin.
-o, --output <path> | Write the ISO to a new output file.
-h, --help | Print help information.

5.2.14. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
You obtained the installation program and generated the Ignition config files for your cluster.

You used the Ignition config files to create RHCOS machines for your cluster.

Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**

1. Monitor the bootstrap process:

   ```
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \  
   --log-level=info
   
   1 For `<installation_directory>`, specify the path to the directory that you stored the
      installation files in.
   
   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.
   
   **Example output**

   INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
   INFO API v1.19.0 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   INFO It is now safe to remove the bootstrap resources
   
   The command succeeds when the Kubernetes API server signals that it has been bootstrapped
   on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   **IMPORTANT**

   You must remove the bootstrap machine from the load balancer at this point.
   You can also remove or reformat the machine itself.

**5.2.15. Logging in to the cluster by using the CLI**

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The
`kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the
correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container
Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   `1`
1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   
   Example output

   system:admin
   ```

### 5.2.16. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

#### Prerequisites

- You added machines to your cluster.

#### Procedure

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   
   Example output

   NAME      STATUS    ROLES   AGE   VERSION
   master-0  Ready     master  63m   v1.19.0
   master-1  Ready     master  63m   v1.19.0
   master-2  Ready     master  64m   v1.19.0
   
   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the `Pending` or `Approved` status for each machine that you added to the cluster:

   ```bash
   $ oc get csr
   
   Example output

   NAME        AGE     REQUESTOR                                                                   CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
   ```
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name>
  ```

  **1**

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:
If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name>
  ```

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
  ```

After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```bash
$ oc get nodes
```

### Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

### 5.2.17. Initial Operator configuration
After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

   ```bash
   $ watch -n5 oc get clusteroperators
   ```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

**5.2.17.1. Image registry removed during installation**
On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE
The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

5.2.17.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

5.2.17.3. Configuring block registry storage

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the Recreate rollout strategy.

IMPORTANT
Block storage volumes are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. To set the image registry storage as a block storage type, patch the registry so that it uses the Recreate rollout strategy and runs with only one (1) replica:

   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
   {
   "rolloutStrategy": "Recreate",
   "replicas": 1
   }}'

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

3. Edit the registry configuration so that it references the correct PVC.

5.2.18. Completing installation on user-provisioned infrastructure
After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   ```bash
   $ watch -n5 oc get clusteroperators
   ```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```bash
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```
1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

INFO Waiting up to 30m0s for the cluster to initialize...

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

Example output

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 1 9m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apisher-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 3m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apisher-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 1m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apisher-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 2m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 0 5m</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace>
```
Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

5.2.19. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

5.2.20. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

5.3. INSTALLING A CLUSTER ON BARE METAL IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.6, you can install a cluster on bare metal infrastructure that you provision in a restricted network.

**IMPORTANT**

While you might be able to follow this procedure to deploy a cluster on virtualized or cloud environments, you must be aware of additional considerations for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you attempt to install an OpenShift Container Platform cluster in such an environment.

5.3.1. Prerequisites

- Create a registry on your mirror host and obtain the imageContentSources data for your version of OpenShift Container Platform.

**IMPORTANT**

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.
• Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.

• Review details about the OpenShift Container Platform installation and update processes.

• If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

  NOTE

  Be sure to also review this site list if you are configuring a proxy.

5.3.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

  IMPORTANT

  Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

5.3.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

• The ClusterVersion status includes an Unable to retrieve available updates error.

• By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

5.3.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

•
- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

5.3.4. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

5.3.4.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines. If you are running a three-node cluster, running zero compute machines is supported. Running one compute machine is not supported.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

5.3.4.2. Network connectivity requirements
All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

5.3.4.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One CPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = CPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

5.3.4.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

5.3.5. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

Prerequisites

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.

Procedure
1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

5.3.5.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 5.28. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Port</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 5.29. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 5.30. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements

The infrastructure that you provision for your cluster must meet the following network topology requirements.

Load balancers

Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

Table 5.31. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
</table>
Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the \texttt{/readyz} endpoint for the API server health check probe.

### Table 5.32. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the \texttt{/readyz} endpoint to the removal of the API server instance from the pool. Within the time frame after \texttt{/readyz} returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

**Table 5.32. Application Ingress load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>
TIP

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources

- Configuring chrony time service

5.3.5.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 5.33. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>
Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.

### Routes

```
*.apps.<cluster_name>.<base_domain>.
```

Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

### Bootstrap

```
bootstrap.<cluster_name>.<base_domain>.
```

Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

### Master hosts

```
<master><n>.<cluster_name>.<base_domain>.
```

DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

### Worker hosts

```
<worker><n>.<cluster_name>.<base_domain>.
```

Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 5.5. Sample DNS zone database**

```
$TTL 1W
@ IN SOA ns1.example.com. root (....)
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

**Example 5.6. Sample DNS zone database for reverse records**

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root (  
 2019070700 ; serial 
 3H ; refresh (3 hours) 
 30M ; retry (30 minutes) 
 2W ; expiry (2 weeks) 
 1W ) ; minimum (1 week) 
IN NS ns1.example.com. 
IN MX 10 smtp.example.com. 
; 
ns1 IN A 192.168.1.5 
smtip IN A 192.168.1.5 
; 
helper IN A 192.168.1.5 
helper.ocp4 IN A 192.168.1.5 
; 
The api identifies the IP of your load balancer. 
api.ocp4 IN A 192.168.1.5 
api-int.ocp4 IN A 192.168.1.5 
; 
The wildcard also identifies the load balancer. 
*.apps.ocp4 IN A 192.168.1.5 
; 
; Create an entry for the bootstrap host. 
bootstrap.ocp4 IN A 192.168.1.96 
; 
; Create entries for the master hosts. 
master0.ocp4 IN A 192.168.1.97 
master1.ocp4 IN A 192.168.1.98 
master2.ocp4 IN A 192.168.1.99 
; 
; Create entries for the worker hosts. 
worker0.ocp4 IN A 192.168.1.11 
worker1.ocp4 IN A 192.168.1.17 
; 
;EOF
```

The syntax is "last octet" and the host must have an FQDN with a trailing dot.

97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
```
5.3.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.
2. Start the **ssh-agent** process as a background task:

```
$ eval "$(ssh-agent -s)"
```

**Example output**

Agent pid 31874

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the **ssh-agent**:  

```
$ ssh-add <path>/<file_name>
```

**Example output**

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

**5.3.7. Manually creating the installation configuration file**

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

- Obtain the **imageContentSources** section from the output of the command to mirror the repository.

- Obtain the contents of the certificate for your mirror registry.

**Procedure**

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```
You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

```
NOTE
You must name this configuration file `install-config.yaml`.

- Unless you use a registry that RHCOS trusts by default, such as `docker.io`, you must provide the contents of the certificate for your mirror repository in the `additionalTrustBundle` section. In most cases, you must provide the certificate for your mirror.

- You must include the `imageContentSources` section from the output of the command to mirror the repository.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

```
IMPORTANT
The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

5.3.7.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

```
NOTE
After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT
The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

5.3.7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 5.34. Required parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
### 5.3.7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 5.35. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>networking</strong></td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>networking.network</strong></td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either <a href="#">OpenShiftSDN</a> or <a href="#">OVN Kubernetes</a>. The default value is <a href="#">OpenShiftSDN</a>.</td>
</tr>
<tr>
<td><strong>networking.clusterNetwork</strong></td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>- networking: serviceNetwork: - 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- networking: machineNetwork: - cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use <code>networking.machineNetwork</code>. An IP address block.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE: Set the <code>networking.machineNetwork</code> to match the CIDR that the preferred NIC resides in.</td>
</tr>
</tbody>
</table>

### 5.3.7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 5.36. Optional parameters**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong> If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere, or</strong>{}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the <strong>x86_64</strong> architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</td>
</tr>
</tbody>
</table>

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

### 5.3.7.2. Sample `install-config.yaml` file for bare metal

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, `-`, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading (SMT), or **hyperthreading**. By default, SMT is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

**NOTE**

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.

**IMPORTANT**

If you disable **hyperthreading**, whether in the BIOS or in the `install-config.yaml`, ensure that your capacity planning accounts for the dramatically decreased machine performance.

You must set the value of the **replicas** parameter to **0**. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines.
for the cluster to use before you finish installing OpenShift Container Platform.

7. The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

8. The cluster name that you specified in your DNS records.

9. A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

10. The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23, then each node is assigned a /23 subnet out of the given `cidr`, which allows for $510 (2^{32 - 23} - 2)$ pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

11. The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

12. You must set the platform to `none`. You cannot provide additional platform configuration variables for your platform.

13. Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

   **IMPORTANT**

   The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

14. For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

15. The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

   **NOTE**

   For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

16. Provide the contents of the certificate file that you used for your mirror registry.

17. Provide the `imageContentSources` section from the output of the command to mirror the repository.
5.3.7.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

NOTE

For bare metal installations, if you do not assign node IP addresses from the range that is specified in the `networking.machineNetwork[].cidr` field in the `install-config.yaml` file, you must include them in the `proxy.noProxy` field.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For
example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

**NOTE**

Only the Proxy object named cluster is supported, and no additional proxies can be created.

5.3.8. Configuring a three-node cluster

You can optionally install and run three-node clusters in OpenShift Container Platform with no workers. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for development, production, and testing.

**Procedure**

- Edit the install-config.yaml file to set the number of compute replicas, which are also known as worker replicas, to 0, as shown in the following compute stanza:

```yaml
compute:
  - name: worker
    platform: {}
    replicas: 0
```

5.3.9. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.

- You created the `install-config.yaml` installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Example output

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

   WARNING

   If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.
## IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become worker nodes.

1. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   c. Save and exit the file.

2. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```
   ├── auth
   │    └── kubeadm-password
   │    └── kubeconfig
   │    └── bootstrap.ign
   │    └── master.ign
   │    └── metadata.json
   │    └── worker.ign
   ```

### 5.3.10. Configuring chrony time service

You must set the time server and related settings used by the chrony time service (`chronyd`) by modifying the contents of the `chrony.conf` file and passing those contents to your nodes as a machine config.

**Procedure**

1. Create the contents of the `chrony.conf` file and encode it as base64. For example:

   ```bash
   $ cat << EOF | base64
   pool 0.rhel.pool.ntp.org iburst
   driftfile /var/lib/chrony/drift
   makestep 1.0 3
   rtsync
   logdir /var/log/chrony
   EOF
   ```
1 Specify any valid, reachable time source, such as the one provided by your DHCP server.

Example output

ICAglHNlcnZlciBjbjG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXlvbgLiL2Nocm9ueS9kcm1lmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2RpclAv
dmFyL2xvZy9jaHJvbmnK

2. Create the MachineConfig object file, replacing the base64 string with the one you just created. This example adds the file to master nodes. You can change it to worker or make an additional MachineConfig for the worker role. Create MachineConfig files for each type of machine that your cluster uses:

```bash
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-masters-chrony-configuration
spec:
  config:
    config: {}  
    security:
      tls: {}
    timeouts: {}
    version: 3.1.0
  networkd: {}
  passwd: {}
  storage:
    files:
      - contents:
          source: data:text/plain;charset=utf-8;base64,ICAglHNlcnZlciBjbjG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXlvbgLiL2Nocm9ueS9kcm1lmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2RpclAv
dmFyL2xvZy9jaHJvbmnK
        mode: 420
        overwrite: true
        path: /etc/chrony.conf
        osImageURL: ""
EOF
```

1 Specify an octal value mode for the mode field in the machine config file. After creating the file and applying the changes, the mode is converted to a decimal value. You can check the YAML file with the command `oc get mc <mc-name> -o yaml`.

3. Make a backup copy of the configuration files.

4. Apply the configurations in one of two ways:

   - If the cluster is not up yet, after you generate manifest files, add this file to the `<installation_directory>/openshift` directory, and then continue to create the cluster.
If the cluster is already running, apply the file:

```
$ oc apply -f ./99-masters-chrony-configuration.yaml
```

### 5.3.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.

**NOTE**

The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and planned for removal in a future release of OpenShift Container Platform 4.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments**: You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the `APPEND` parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special `coreos.inst.*` arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.

- **Ignition configs**: OpenShift Container Platform Ignition config files (`*ign`) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the `coreos-installer` to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.

- **coreos-installer**: You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the `coreos-installer` command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.
NOTE

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

5.3.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

Prerequisites

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.

Procedure

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   IMPORTANT

   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the RHCOS image mirror page.

   IMPORTANT

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

   ISO file names resemble the following example:

   `rhcos-<version>-live.<architecture>.iso`

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:

   - Burn the ISO image to a disk and boot it directly.
   - Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.
5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the coreos-installer.

6. Run the coreos-installer command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

   ```
   $ sudo coreos-installer install \
   --ignition-url=https://host/worker.ign /dev/sda
   ```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.

   **IMPORTANT**

   You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

5.3.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Configure suitable PXE or iPXE infrastructure.
- Have access to an HTTP server that you can access from your computer.

**Procedure**

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**

   You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS kernel, initramfs and rootfs files from the RHCOS image mirror page.
The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate kernel, initramfs, and rootfs artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel**: rhcos-<version>-live-kernel-<architecture>
- **initramfs**: rhcos-<version>-live-initramfs.<architecture>.img
- **rootfs**: rhcos-<version>-live-rootfs.<architecture>.img

3. Upload the additional files that are required for your booting method:

- For traditional PXE, upload the kernel and initramfs files to your TFTP server and the rootfs file to your HTTP server.
- For iPXE, upload the kernel, initramfs, and rootfs files to your HTTP server.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images.

Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- For PXE:

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> ①
```

① Specify the location of the live kernel file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

② If you use multiple NICs, specify a single interface in the ip option. For example, to use DHCP on a NIC that is named eno1, set `ip=eno1:dhcp`. 
Specify locations of the RHCOS files that you uploaded to your HTTP server. The `initrd` parameter value is the location of the `initramfs` file, the `coreos.live.rootfs_url` parameter value is the location of the `rootfs` file, and the `coreos.inst.ignition_url` parameter value is the location of the `bootstrap Ignition config` file.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `APPEND` line. For example, add `console=tt00 console=tt0S0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?.

- For iPXE:

```sh

1 Specify locations of the RHCOS files that you uploaded to your HTTP server. The `kernel` parameter value is the location of the `kernel` file, the `initrd=main` argument is needed for booting on UEFI systems, the `coreos.live.rootfs_url` parameter value is the location of the `rootfs` file, and the `coreos.inst.ignition_url` parameter value is the location of the `bootstrap Ignition config` file.

2 If you use multiple NICs, specify a single interface in the `ip` option. For example, to use DHCP on a NIC that is named `eno1`, set `ip=eno1:dhcp`.

3 Specify the location of the `initramfs` file that you uploaded to your HTTP server.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `kernel` line. For example, add `console=tt00 console=tt0S0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?.

6. If you use PXE UEFI, perform the following actions:

   a. Provide the `shimx64.efi` and `grubx64.efi` EFI binaries and the `grub.cfg` file that are required for booting the system.

   - Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the `images/efiboot.img` file to your host:

```sh
$ mkdir -p /mnt/iso
```

CHAPTER 5. INSTALLING ON BARE METAL
From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
$ umount /mnt/efiboot
$ umount /mnt/iso
```

- From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
$ umount /mnt/efiboot
$ umount /mnt/iso
```

- Copy the `EFI/redhat/grub.cfg` file that is included in the RHCOS ISO to your TFTP server.

b. Edit the `grub.cfg` file to include arguments similar to the following:

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
   linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
   coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
   initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

where:

```
rhcos-<version>-live-kernel-<architecture>
   Specifies the kernel file that you uploaded to your TFTP server.
http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
   Specifies the location of the live rootfs image that you uploaded to your HTTP server.
http://<HTTP_server>/bootstrap.ign
   Specifies the location of the bootstrap Ignition config file that you uploaded to your HTTP server.
rhcos-<version>-live-initramfs.<architecture>.img
   Specifies the location of the initramfs file that you uploaded to your TFTP server.
```

**NOTE**

For more information on how to configure a PXE server for UEFI boot, see the Red Hat Knowledgebase article: How to configure/setup a PXE server for UEFI boot for Red Hat Enterprise Linux?
7. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

### 5.3.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running `coreos-installer` manually from the live system
- Embedding Ignition configs in an ISO

The advanced configuration topics for manual Red Hat Enterprise Linux CoreOS (RHCOS) installations detailed in this section relate to disk partitioning, networking, and using Ignition configs in different ways.

### 5.3.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

**Procedure**

1. Boot the ISO installer.

2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as `nmcli` or `nmtui`.

3. Run the `coreos-installer` command to install the system, adding the `--copy-network` option to copy networking configuration. For example:
The `--copy-network` option only copies networking configuration found under
`/etc/NetworkManager/system-connections`. In particular, it does not copy the
system hostname.

4. Reboot into the installed system.

5.3.11.3.2. Disk partitioning

The disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat
Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the
same partition layout, unless the default partitioning configuration is overridden. During the RHCOS
installation, the size of the root file system is increased to use the remaining available space on the
target device.

However, there are two cases where you might want to intervene to override the default partitioning
when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add
  separate storage to a partition. This is officially supported for making `/var` or a subdirectory of
  `/var`, such as `/var/lib/etcd`, a separate partition, but not both.

  **IMPORTANT**
  Kubernetes supports only two filesystem partitions. If you add more than one
  partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift
  Container Platform on an existing node and want to retain data partitions installed from your
  previous operating system, there are both boot arguments and options to `coreos-installer` that
  allow you to retain existing data partitions.

5.3.11.3.2.1. Creating a separate `/var` partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However,
there are cases where you might want to create separate partitions in a part of the filesystem that you
expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the
`/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and
  containers are added to a system.

- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as
  performance optimization of etcd storage.

- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as
needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadmin-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...
   ```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the worker systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
       machineconfiguration.openshift.io/role: worker
   name: 98-var-partition
   spec:
     config:
       ignition:
         version: 3.1.0
     storage:
       disks:
         - device: /dev/<device_name> ①
           partitions:
             - label: var
               startMiB: <partition_start_offset> ②
               sizeMiB: <partition_size> ③
           filesystems:
             - device: /dev/disk/by-partlabel/var
               path: /var
               format: xfs
       systemd:
           units:
   ```
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.

The prjquota mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

   ```
   $ openshift-install create ignition-configs --dir $HOME/clusterconfig
   $ ls $HOME/clusterconfig/
   auth  bootstrap.ign  master.ign  metadata.json  worker.ign
   ```

Now you can use the Ignition config files as input to the ISO or PXE manual installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 5.3.11.3.2.2. Retaining existing partitions

For an ISO installation, you can add options to the coreos-installer command line that causes the installer to maintain one or more existing partitions. For a PXE installation, you can APPEND coreos.inst.* options to preserve partitions.

Saved partitions might be partitions from an existing OpenShift Container Platform system that has data partitions that you want to keep. Here are a few tips:
If you save existing partitions, and those partitions do not leave enough space for RHCOs, installation will fail without damaging the saved partitions.

- Identify the disk partitions you want to keep either by partition label or by number.

**For an ISO installation**

This example preserves any partition in which the partition label begins with `data` (*data*):

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the `coreos-installer` in a way that preserves the sixth (6) partition on the disk:

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partindex 5- /dev/sda
```

In the previous examples where partition saving is used, `coreos-installer` recreates the partition immediately.

**For a PXE installation**

This `APPEND` option preserves any partition in which the partition label begins with ‘data’ (*data*):

```bash
coreos.inst.save_partlabel=data*
```

This `APPEND` option preserves partitions 5 and higher:

```bash
coreos.inst.save_partindex=5-
```

This `APPEND` option preserves partition 6:

```bash
coreos.inst.save_partindex=6
```

5.3.11.3.3. Identifying Ignition configs

When doing an RHCOs manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config**: Every manual RHCOs installation needs to pass one of the Ignition config files generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, to carry out the installation.

  **IMPORTANT**

  It is not recommended to modify these files.

  For PXE installations, you pass the Ignition configs on the `APPEND` line using the
**coreos.inst.ignition_url** option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the **coreos-installer** command line with the **--ignition-url** option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config:** This type must be created manually and should be avoided if possible, as it is not supported by Red Hat. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before and/or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be performed once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

  For PXE or ISO boots, you can create the Ignition config and **APPEND** the **ignition.config.url** option to identify the location of the Ignition config. You also need to append **ignition.firstboot** **ignition.platform.id=metal** or the **ignition.config.url** option will be ignored.

### 5.3.11.3.3.1. Embedding an Ignition config in the RHCOS ISO

You can embed a live install Ignition config directly in an RHCOS ISO image. When the ISO image is booted, the embedded config will be applied automatically.

**Procedure**

1. Download the **coreos-installer** binary from the following image mirror page:
   

2. Retrieve the RHCOS ISO image and the Ignition config file, and copy them into an accessible directory, such as `/mnt`:

   ```bash
   # cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
   # chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
   ```

3. Run the following command to embed the Ignition config into the ISO:

   ```bash
   # ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
   /mnt/rhcos-<version>-live.x86_64.iso
   ```

   You can now use that ISO to install RHCOS using the specified live install Ignition config.

   **IMPORTANT**

   Using **coreos-installer iso ignition embed** to embed a file generated by **openshift-installer**, such as **bootstrap.ign**, **master.ign** and **worker.ign**, is unsupported and not recommended.

4. To show the contents of the embedded Ignition config and direct it into a file, run:

   ```bash
   # ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
   # diff -s bootstrap.ign mybootstrap.ign
   ```

   **Example output**

   Files bootstrap.ign and mybootstrap.ign are identical.
5. To remove the Ignition config and return the ISO to its pristine state so you can reuse it, run:

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

You can now embed another Ignition config into the ISO or use the ISO in its pristine state.

5.3.11.3.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

Routing and bonding options at RHCOS boot prompt

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

**IMPORTANT**

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>To configure an IP address, either use DHCP (ip=dhcp) or set an individual static IP address (ip=(&lt;host_ip&gt;)). Then identify the DNS server IP address (nameserver=(&lt;dns_ip&gt;)) on each node. This example sets:</td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41</td>
</tr>
<tr>
<td>● The node’s IP address to 10.10.10.2</td>
<td></td>
</tr>
<tr>
<td>● The gateway address to 10.10.10.254</td>
<td></td>
</tr>
<tr>
<td>● The netmask to 255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>● The hostname to core0.example.com</td>
<td></td>
</tr>
<tr>
<td>● The DNS server address to 4.4.4.41</td>
<td></td>
</tr>
<tr>
<td>Specify multiple network interfaces by specifying multiple ip= entries.</td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>Optional: You can configure routes to additional networks by setting an rd.route= value.</td>
<td>To configure the default gateway:</td>
</tr>
<tr>
<td>If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.</td>
<td>ip=::10.10.10.254:::</td>
</tr>
<tr>
<td>To configure the route for the additional network:</td>
<td>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</td>
</tr>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used.</td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=:::core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>You can combine DHCP and static IP configurations on systems with multiple network interfaces.</td>
<td>ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Optional: You can configure VLANs on individual interfaces by using the <code>vlan=</code> parameter.</td>
<td>To configure a VLAN on a network interface and use a static IP address:</td>
</tr>
<tr>
<td></td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0</td>
</tr>
<tr>
<td></td>
<td>To configure a VLAN on a network interface and to use DHCP:</td>
</tr>
<tr>
<td></td>
<td>ip=enp2s0.100:dhcp</td>
</tr>
<tr>
<td></td>
<td>vlan=enp2s0.100:enp2s0</td>
</tr>
<tr>
<td>You can provide multiple DNS servers by adding a <code>nameserver=</code> entry for each server.</td>
<td>nameserver=1.1.1.1</td>
</tr>
<tr>
<td></td>
<td>nameserver=8.8.8.8</td>
</tr>
<tr>
<td>Optional: Bonding multiple network interfaces to a single interface is supported using the <code>bond=</code> option. In these two examples:</td>
<td>To configure the bonded interface to use DHCP, set the bond’s IP address to <code>dhcp</code>. For example:</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp</td>
</tr>
<tr>
<td></td>
<td>To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none</td>
</tr>
</tbody>
</table>
Optional: You can configure VLANs on bonded interfaces by using the `vlan=` parameter.

To configure the bonded interface with a VLAN and to use DHCP:

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

To configure the bonded interface with a VLAN and to use a static IP address:

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example:

- The syntax for configuring a team interface is: `team=name[:network_interfaces]`
  - `name` is the team device name (team0) and `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (`em1`, `em2`).

**NOTE**

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this Red Hat Knowledgebase Article.

To configure a network team:

```
team=team0:em1,em2
ip=team0:dhcp
```

coreos.inst boot options for ISO or PXE install

While you can pass most standard installation boot arguments to the live installer, there are several arguments that are specific to the RHCOS live installer.

- For ISO, these options can be added by interrupting the RHCOS installer.

- For PXE or iPXE, these options must be added to the `APPEND` line before starting the PXE kernel. You cannot interrupt a live PXE install.

The following table shows the RHCOS live installer boot options for ISO and PXE installs.

**Table 5.37. coreos.inst boot options**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
</table>

---

1152
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreos.inst.install_dev</td>
<td>Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda, although sda is allowed.</td>
</tr>
<tr>
<td>coreos.inst.ignition_url</td>
<td>Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded.</td>
</tr>
<tr>
<td>coreos.inst.save_partlabel</td>
<td>Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.save_partindex</td>
<td>Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.insecure</td>
<td>Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.</td>
</tr>
<tr>
<td>coreos.inst.image_url</td>
<td>Optional: Download and install the specified RHCOS image.</td>
</tr>
<tr>
<td></td>
<td>- This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
<tr>
<td></td>
<td>- While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install.</td>
</tr>
<tr>
<td></td>
<td>- If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform.</td>
</tr>
<tr>
<td></td>
<td>- Only HTTP and HTTPS protocols are supported.</td>
</tr>
<tr>
<td>coreos.inst.skip_reboot</td>
<td>Optional: The system will not reboot after installing. Once the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
</tbody>
</table>
coreos.inst.platform_id

Optional: The Ignition platform ID of the platform the RHCOS image is being installed on. Default is metal. This option determines whether or not to request an Ignition config from the cloud provider, such as VMware. For example:

```
coreos.inst.platform_id=vmware
```

ignition.config.url

Optional: The URL of the Ignition config for the live boot. For example, this can be used to customize how `coreos-installer` is invoked, or to run code before or after the installation. This is different from `coreos.inst.ignition_url`, which is the Ignition config for the installed system.

coreos-installer options for ISO install

You can also install RHCOS by invoking the `coreos-installer` command directly from the command line. The kernel arguments in the previous table provide a shortcut for automatically invoking `coreos-installer` at boot time, but you can pass similar arguments directly to `coreos-installer` when running it from a shell prompt.

The following table shows the options and subcommands you can pass to the `coreos-installer` command from a shell prompt during a live install.

Table 5.38. `coreos-installer` command-line options, arguments, and subcommands

<table>
<thead>
<tr>
<th>Command-line options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-u, --image-url &lt;url&gt;</code></td>
<td>Specify the image URL manually.</td>
</tr>
<tr>
<td><code>-f, --image-file &lt;path&gt;</code></td>
<td>Specify a local image file manually.</td>
</tr>
<tr>
<td><code>-i, --ignition-file &lt;path&gt;</code></td>
<td>Embed an Ignition config from a file.</td>
</tr>
<tr>
<td><code>-l, --ignition-url &lt;URL&gt;</code></td>
<td>Embed an Ignition config from a URL.</td>
</tr>
<tr>
<td><code>--ignition-hash &lt;digest&gt;</code></td>
<td>Digest type-value of the Ignition config.</td>
</tr>
<tr>
<td><code>-p, --platform &lt;name&gt;</code></td>
<td>Override the Ignition platform ID.</td>
</tr>
<tr>
<td><code>--append-karg &lt;arg&gt;...</code></td>
<td>Append the default kernel argument.</td>
</tr>
<tr>
<td><code>--delete-karg &lt;arg&gt;...</code></td>
<td>Delete the default kernel argument.</td>
</tr>
</tbody>
</table>
-n, --copy-network

Copy the network configuration from the install environment.

**IMPORTANT**

The `--copy-network` option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

|--network-dir <path> For use with `-n`. Default is `/etc/NetworkManager/system-connections/`.

|--save-partlabel <lx>.. Save partitions with this label glob.

|--save-partindex <id>... Save partitions with this number or range.

|--offline Force offline installation.

|--insecure Skip signature verification.

|--insecure-ignition Allow Ignition URL without HTTPS or hash.

|--architecture <name> Target CPU architecture. Default is `x86_64`.

|--preserve-on-error Do not clear partition table on error.

-h, --help Print help information.

**Command-line argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;device&gt;</td>
<td>The destination device.</td>
</tr>
</tbody>
</table>

**coreos-installer embedded Ignition commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ coreos-installer iso ignition embed &lt;options&gt; --ignition-file &lt;file_path&gt; &lt;ISO_image&gt;</code></td>
<td>Embed an Ignition config in an ISO image.</td>
</tr>
<tr>
<td><code>coreos-installer iso ignition show &lt;options&gt; &lt;ISO_image&gt;</code></td>
<td>Show the embedded Ignition config from an ISO image.</td>
</tr>
</tbody>
</table>
Remove the embedded Ignition config from an ISO image.

**coreos-installer ISO Ignition options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f, --force</td>
<td>Overwrite an existing Ignition config.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

**coreos-installer PXE Ignition commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreos-installer pxe ignition wrap &lt;options&gt;</td>
<td>Wrap an Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;image_name&gt;</td>
<td>Show the wrapped Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;initrd_name&gt;</td>
<td>Show the wrapped Ignition config in an initrd image.</td>
</tr>
</tbody>
</table>

**coreos-installer PXE Ignition options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

5.3.12. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
You obtained the installation program and generated the Ignition config files for your cluster.

You used the Ignition config files to create RHCOS machines for your cluster.

Procedure

1. Monitor the bootstrap process:

   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \  
   --log-level=info

   For `<installation_directory>`, specify the path to the directory that you stored the 
   installation files in.

   To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   Example output

   INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
   INFO API v1.19.0 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   INFO It is now safe to remove the bootstrap resources

   The command succeeds when the Kubernetes API server signals that it has been bootstrapped 
   on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   IMPORTANT

   You must remove the bootstrap machine from the load balancer at this point. 
   You can also remove or reformat the machine itself.

5.3.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

**Example output**

```
system:admin
```

### 5.3.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

**Example output**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

```
NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
```
CHAPTER 5. INSTALLING ON BARE METAL

csr-8vnps 15m system:serviceaccount:openshift-machine-config-operator:nodebootstrapper Pending
...
In this example, two machines are joining the cluster. You might see more approved CSRs in the
list.
3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in
Pending status, approve the CSRs for your cluster machines:

NOTE
Because the CSRs rotate automatically, approve your CSRs within an hour of
adding the machines to the cluster. If you do not approve them within an hour, the
certificates will rotate, and more than two certificates will be present for each
node. You must approve all of these certificates. Once the client CSR is
approved, the Kubelet creates a secondary CSR for the serving certificate, which
requires manual approval. Then, subsequent serving certificate renewal requests
are automatically approved by the machine-approver if the Kubelet requests a
new certificate with identical parameters.

NOTE
For clusters running on platforms that are not machine API enabled, such as bare
metal and other user-provisioned infrastructure, you must implement a method
of automatically approving the kubelet serving certificate requests (CSRs). If a
request is not approved, then the oc exec, oc rsh, and oc logs commands
cannot succeed, because a serving certificate is required when the API server
connects to the kubelet. Any operation that contacts the Kubelet endpoint
requires this certificate approval to be in place. The method must watch for new
CSRs, confirm that the CSR was submitted by the node-bootstrapper service
account in the system:node or system:admin groups, and confirm the identity
of the node.
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

To approve all pending CSRs, run the following command:
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve

NOTE
Some Operators might not become available until some CSRs are approved.
4. Now that your client requests are approved, you must review the server requests for each
machine that you added to the cluster:

1159


$ oc get csr

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:
  
  $ oc adm certificate approve <csr_name>  

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:
  
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve

6. After all client and server CSRs have been approved, the machines have the Ready status. Verify this by running the following command:

$ oc get nodes

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the Ready status.

Additional information

- For more information on CSRs, see Certificate Signing Requests.

**5.3.15. Initial Operator configuration**
After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**
- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

   ```bash
   $ watch -n5 oc get clusteroperators
   ``

**Example output**

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED 
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      3h56m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      4h30m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      3h56m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      3h56m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. Configure the Operators that are not available.

**5.3.15.1. Disabling the default OperatorHub sources**
Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```bash
  $ oc patch OperatorHub cluster --type json \
  -p '[["op": "add", "path": "/spec/disableAllDefaultSources", "value": true]]'
  ``

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

5.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

5.3.15.2.1. Changing the image registry’s management state

To start the image registry, you must change the Image Registry Operator configuration’s `managementState` from `Removed` to `Managed`.

**Procedure**

- Change `managementState` Image Registry Operator configuration from `Removed` to `Managed`. For example:

  ```bash
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": "managementState": "Managed"}’
  ```

5.3.15.2.2. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.
• Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

• Must have 100Gi capacity.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   ``

   **NOTE**

   If the storage type is **emptyDIR**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   ``

   **Example output**

   ```yaml
   storage:
   pvc:
   claim:
   ``

   Leave the `claim` field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

   ```bash
   $ oc get clusteroperator image-registry
   ``

5. Ensure that your registry is set to managed to enable building and pushing of images.

   • Run:

   ```bash
   $ oc edit configs.imageregistry/cluster
   Then, change the line
   ```
managementState: Removed
to
managementState: Managed

5.3.15.2.3. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

  ```
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"storage":{"emptyDir":{}}}'}
  ```

  **WARNING**
  Configure this option for only non-production clusters.

  If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

  ```
  Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
  ```

  Wait a few minutes and run the command again.

5.3.15.2.4. Configuring block registry storage

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only one (1) replica:

  ```
  $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
  {"rolloutStrategy":"Recreate","replicas":1}}'
  ```
2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

3. Edit the registry configuration so that it references the correct PVC.

5.3.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

   ```
   $ watch -n5 oc get clusteroperators
   ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

**Example output**

```
NAMESPACE                NAME                                            READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator openshift-apiserver-operator-85cb746d55-zqhs8  1/1 Running 0
  Running     0
3m
1m
1m
```

OpenShift Container Platform 4.6 Installing
b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace>
```

Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Register your cluster on the Cluster registration page.

**5.3.17. Telemetry access for OpenShift Container Platform**

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

**5.3.18. Next steps**

- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the must-gather tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can opt out of remote health reporting.
CHAPTER 6. DEPLOYING INSTALLER-PROVISIONED CLUSTERS ON BARE METAL

6.1. OVERVIEW

Installer-provisioned installation provides support for installing OpenShift Container Platform on bare metal nodes. This guide provides a methodology to achieving a successful installation.

During installer-provisioned installation on bare metal, the installer on the bare metal node labeled as provisioner creates a bootstrap VM. The role of the bootstrap VM is to assist in the process of deploying an OpenShift Container Platform cluster. The bootstrap VM connects to the baremetal network and to the provisioning network, if present, via the network bridges.

When the installation of OpenShift Container Platform control plane nodes is complete and fully operational, the installer destroys the bootstrap VM automatically and moves the virtual IP addresses (VIPs) to the appropriate nodes accordingly. The API VIP moves to the control plane nodes and the Ingress VIP moves to the worker nodes.

6.2. PREREQUISITES

Installer-provisioned installation of OpenShift Container Platform requires:

1. One provisioner node with Red Hat Enterprise Linux (RHEL) 8.x installed.
2. Three control plane nodes.
3. Baseboard Management Controller (BMC) access to each node.
4. At least one network:
   a. One required routable network
b. One optional network for provisioning nodes; and,
c. One optional management network.

Before starting an installer-provisioned installation of OpenShift Container Platform, ensure the hardware environment meets the following requirements.

### 6.2.1. Node requirements

Installer-provisioned installation involves a number of hardware node requirements:

- **CPU architecture:** All nodes must use \texttt{x86_64} CPU architecture.
- **Similar nodes:** Red Hat recommends nodes have an identical configuration per role. That is, Red Hat recommends nodes be the same brand and model with the same CPU, memory, and storage configuration.
- **Baseboard Management Controller:** The provisioner node must be able to access the baseboard management controller (BMC) of each OpenShift Container Platform cluster node. You may use IPMI, RedFish, or a proprietary protocol.
- **Latest generation:** Nodes must be of the most recent generation. Because the installer-provisioned installation relies on BMC protocols, the hardware must support IPMI cipher suite 17. Additionally, RHEL 8 ships with the most recent drivers for RAID controllers. Ensure that the nodes are recent enough to support RHEL 8 for the provisioner node and RHCOS 8 for the control plane and worker nodes.
- **Registry node:** Optional: If setting up a disconnected mirrored registry, it is recommended the registry reside in its own node.
- **Provisioner node:** Installer-provisioned installation requires one provisioner node.
- **Control plane:** Installer-provisioned installation requires three control plane nodes for high availability.
- **Worker nodes:** While not required, a typical production cluster has one or more worker nodes. Smaller clusters are more resource efficient for administrators and developers during development and testing.
- **Network interfaces:** Each node must have at least one network interface for the routable baremetal network. Each node must have one network interface for a provisioning network when using the provisioning network for deployment. Using the provisioning network is the default configuration. Network interface naming must be consistent across control plane nodes for the provisioning network. For example, if a control plane node uses the eth0 NIC for the provisioning network, the other control plane nodes must use it as well.
- **Unified Extensible Firmware Interface (UEFI):** Installer-provisioned installation requires UEFI boot on all OpenShift Container Platform nodes when using IPv6 addressing on the provisioning network. In addition, UEFI Device PXE Settings must be set to use the IPv6 protocol on the provisioning network NIC, but omitting the provisioning network removes this requirement.

### 6.2.2. Network requirements

Installer-provisioned installation of OpenShift Container Platform involves several network requirements by default. First, installer-provisioned installation involves a non-routable provisioning network.
network for provisioning the operating system on each bare metal node and a routable baremetal network. Since installer-provisioned installation deploys ironic-dnsmasq, the networks should have no other DHCP servers running on the same broadcast domain. Network administrators must reserve IP addresses for each node in the OpenShift Container Platform cluster.

Network Time Protocol (NTP)

It is recommended that each OpenShift Container Platform node in the cluster have access to a Network Time Protocol (NTP) server that is discoverable using DHCP. While installation without an NTP server is possible, asynchronous server clocks can cause errors. Using an NTP server can prevent this issue.

Configuring NICs

OpenShift Container Platform deploys with two networks:

- **provisioning**: The provisioning network is an optional non-routable network used for provisioning the underlying operating system on each node that is a part of the OpenShift Container Platform cluster. When deploying using the provisioning network, the first NIC on each node, such as eth0 or eno1, must interface with the provisioning network.

- **baremetal**: The baremetal network is a routable network. When deploying using the provisioning network, the second NIC on each node, such as eth1 or eno2, must interface with the baremetal network. When deploying without a provisioning network, you can use any NIC on each node to interface with the baremetal network.

**IMPORTANT**

Each NIC should be on a separate VLAN corresponding to the appropriate network.

Configuring the DNS server

Clients access the OpenShift Container Platform cluster nodes over the baremetal network. A network administrator must configure a subdomain or subzone where the canonical name extension is the cluster name.

```
<cluster-name>.<domain-name>
```

For example:

```
test-cluster.example.com
```

Reserving IP addresses for nodes with the DHCP server

For the baremetal network, a network administrator must reserve a number of IP addresses, including:

1. Two virtual IP addresses.
   - One IP address for the API endpoint
   - One IP address for the wildcard Ingress endpoint

2. One IP address for the provisioner node.

3. One IP address for each control plane (master) node.

4. One IP address for each worker node, if applicable.
The following table provides an exemplary embodiment of fully-qualified domain names. The API and Nameserver addresses begin with canonical name extensions. The hostnames of the control plane and worker nodes are exemplary, so you can use any host naming convention you prefer.

<table>
<thead>
<tr>
<th>Usage</th>
<th>Hostname</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>api.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Ingress LB (apps)</td>
<td>*.apps.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Provisioner node</td>
<td>provisioner.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Master-0</td>
<td>openshift-master-0.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Master-1</td>
<td>openshift-master-1.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Master-2</td>
<td>openshift-master-2.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Worker-0</td>
<td>openshift-worker-0.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Worker-1</td>
<td>openshift-worker-1.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
<tr>
<td>Worker-n</td>
<td>openshift-worker-n.&lt;cluster-name&gt;.&lt;domain&gt;</td>
<td>&lt;ip&gt;</td>
</tr>
</tbody>
</table>

Additional requirements with no provisioning network

All installer-provisioned installations require a baremetal network. The baremetal network is a routable network used for external network access to the outside world. In addition to the IP address supplied to the OpenShift Container Platform cluster node, installations without a provisioning network require the following:

- Setting an available IP address from the baremetal network to the bootstrapProvisioningIP configuration setting within the install-config.yaml configuration file.
- Setting an available IP address from the baremetal network to the provisioningHostIP configuration setting within the install-config.yaml configuration file.
- Deploying the OpenShift Container Platform cluster using RedFish Virtual Media/iDRAC Virtual Media.

NOTE

Configuring additional IP addresses for bootstrapProvisioningIP and provisioningHostIP is not required when using a provisioning network.

Port access for the out-of-band management IP address

The out-of-band management IP address is on a separate network from the node. To ensure that the out-of-band management can communicate with the baremetal node during installation, the out-of-band management IP address address must be granted access to the TCP 6180 port.
6.2.3. Configuring nodes

Configuring nodes when using the provisioning network

Each node in the cluster requires the following configuration for proper installation.

**WARNING**

A mismatch between nodes will cause an installation failure.

While the cluster nodes can contain more than two NICs, the installation process only focuses on the first two NICs:

<table>
<thead>
<tr>
<th>NIC</th>
<th>Network</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIC1</td>
<td>provisioning</td>
<td>&lt;provisioning-vlan&gt;</td>
</tr>
<tr>
<td>NIC2</td>
<td>baremetal</td>
<td>&lt;baremetal-vlan&gt;</td>
</tr>
</tbody>
</table>

NIC1 is a non-routable network (**provisioning**) that is only used for the installation of the OpenShift Container Platform cluster.

The Red Hat Enterprise Linux (RHEL) 8.x installation process on the provisioner node might vary. To install Red Hat Enterprise Linux (RHEL) 8.x using a local Satellite server or a PXE server, PXE-enable NIC2.

<table>
<thead>
<tr>
<th>PXE</th>
<th>Boot order</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIC1 PXE-enabled <strong>provisioning</strong> network</td>
<td>1</td>
</tr>
<tr>
<td>NIC2 <strong>baremetal</strong> network. PXE-enabled is optional.</td>
<td>2</td>
</tr>
</tbody>
</table>

**NOTE**

Ensure PXE is disabled on all other NICs.

Configure the control plane and worker nodes as follows:

<table>
<thead>
<tr>
<th>PXE</th>
<th>Boot order</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIC1 PXE-enabled (provisioning network)</td>
<td>1</td>
</tr>
</tbody>
</table>

Configuring nodes without the provisioning network
The installation process requires one NIC:

<table>
<thead>
<tr>
<th>NIC</th>
<th>Network</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>NICx</td>
<td>baremetal</td>
<td>&lt;baremetal-vlan&gt;</td>
</tr>
</tbody>
</table>

NICx is a routable network (baremetal) that is used for the installation of the OpenShift Container Platform cluster, and routable to the Internet.

### 6.2.4. Out-of-band management

Nodes will typically have an additional NIC used by the Baseboard Management Controllers (BMCs). These BMCs must be accessible from the provisioner node.

Each node must be accessible via out-of-band management. When using an out-of-band management network, the provisioner node requires access to the out-of-band management network for a successful OpenShift Container Platform 4 installation.

The out-of-band management setup is out of scope for this document. We recommend setting up a separate management network for out-of-band management. However, using the provisioning network or the baremetal network are valid options.

### 6.2.5. Required data for installation

Prior to the installation of the OpenShift Container Platform cluster, gather the following information from all cluster nodes:

- Out-of-band management IP
  - Examples
    - Dell (iDRAC) IP
    - HP (iLO) IP

When using the provisioning network

- NIC1 (provisioning) MAC address
- NIC2 (baremetal) MAC address

When omitting the provisioning network

- NICx (baremetal) MAC address

### 6.2.6. Validation checklist for nodes

When using the provisioning network

- NIC1 VLAN is configured for the provisioning network.
- NIC2 VLAN is configured for the baremetal network.
NIC1 is PXE-enabled on the provisioner, control plane (master), and worker nodes.

PXE has been disabled on all other NICs.

Control plane and worker nodes are configured.

All nodes accessible via out-of-band management.

A separate management network has been created. (optional)

Required data for installation.

When omitting the provisioning network

NICx VLAN is configured for the baremetal network.

Control plane and worker nodes are configured.

All nodes accessible via out-of-band management.

A separate management network has been created. (optional)

Required data for installation.

6.3. SETTING UP THE ENVIRONMENT FOR AN OPENSFiTH INSTALLATION

6.3.1. Installing RHEL on the provisioner node

With the networking configuration complete, the next step is to install RHEL 8.x on the provisioner node. The installer uses the provisioner node as the orchestrator while installing the OpenShift Container Platform cluster. For the purposes of this document, installing RHEL on the provisioner node is out of scope. However, options include but are not limited to using a RHEL Satellite server, PXE, or installation media.

6.3.2. Preparing the provisioner node for OpenShift Container Platform installation

Perform the following steps to prepare the environment.

Procedure

1. Log in to the provisioner node via ssh.

2. Create a non-root user (kni) and provide that user with sudo privileges.

   # useradd kni
   # passwd kni
   # echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
   # chmod 0440 /etc/sudoers.d/kni

3. Create an ssh key for the new user.

   # su - kni -c "ssh-keygen -t ed25519 -f /home/kni/.ssh/id_rsa -N ""
4. Log in as the new user on the provisioner node.

   $ su - kni

5. Use Red Hat Subscription Manager to register the provisioner node.

   $ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
   $ sudo subscription-manager repos --enable=rhel-8-for-x86_64-appstream-rpms --
   enable=rhel-8-for-x86_64-baseos-rpms

   NOTE
   For more information about Red Hat Subscription Manager, see Using and
   Configuring Red Hat Subscription Manager.

6. Install the following packages.

   $ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool

7. Modify the user to add the libvirt group to the newly created user.

   $ sudo usermod --append --groups libvirt <user>

8. Restart firewalld and enable the http service.

   $ sudo systemctl start firewalld
   $ sudo firewall-cmd --zone=public --add-service=http --permanent
   $ sudo firewall-cmd --reload

9. Start and enable the libvirtd service.

   $ sudo systemctl enable libvirtd --now

10. Create the default storage pool and start it.

    $ sudo virsh pool-define-as --name default --type dir --target /var/lib/libvirt/images
    $ sudo virsh pool-start default
    $ sudo virsh pool-autostart default

11. Configure networking.

    NOTE
    This step can also be run from the web console.

    $ export PUB_CONN=<baremetal_nic_name>
    $ export PROV_CONN=<prov_nic_name>
    $ sudo nohup bash -c "
        nmcli con down "$PROV_CONN"
        nmcli con down "$PUB_CONN"
    "

    CHAPTER 6. DEPLOYING INSTALLER-PROVISIONED CLUSTERS ON BARE METAL
NOTE

The ssh connection might disconnect after executing this step.

The IPv6 address can be any address as long as it is not routable via the baremetal network.

Ensure that UEFI is enabled and UEFI PXE settings are set to the IPv6 protocol when using IPv6 addressing.

12. Configure the IPv4 address on the provisioning network connection.

   $ nmcli connection modify provisioning ipv4.addresses 172.22.0.254/24 ipv4.method manual

13. ssh back into the provisioner node (if required).

   # ssh kni@provisioner.<cluster-name>.<domain>

14. Verify the connection bridges have been properly created.

   $ sudo nmcli con show

   NAME       UUID                                  TYPE      DEVICE
   baremetal  4d5133a5-8351-4bb9-bfd4-3af264801530  bridge    baremetal
   provisioning 43942805-017f-4d7d-a2c2-7cb3324482ed  bridge    provisioning
   virbr0     d9bca40f-eee1-410b-8879-a2d4bb0465e7  bridge    virbr0
   bridge-slave-eno1 76a8ed50-c7e5-4999-b4f6-6d9014dd0812 ethernet eno1
   bridge-slave-eno2 f31c3353-54b7-48de-893a-02d2b34c4736 ethernet eno2

15. Create a pull-secret.txt file.

   $ vim pull-secret.txt

In a web browser, navigate to Install on Bare Metal with user-provisioned infrastructure, and scroll down to the Downloads section. Click Copy pull secret Paste the contents into the pull-secret.txt file and save the contents in the kni user’s home directory.

6.3.3. Retrieving the OpenShift Container Platform installer
Use the latest-4.x version of the installer to deploy the latest generally available version of OpenShift Container Platform:

```
$ export VERSION=latest-4.6
export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print $3}')
```

Additional resources

- See OpenShift Container Platform upgrade channels and releases for an explanation of the different release channels.

### 6.3.4. Extracting the OpenShift Container Platform installer

After retrieving the installer, the next step is to extract it.

**Procedure**

1. Set the environment variables:

   ```
   $ export cmd=openshift-baremetal-install
   $ export pullsecret_file=~/.pull-secret.txt
   $ export extract_dir=$(pwd)
   ```

2. Get the oc binary:

   ```
   ```

3. Extract the installer:

   ```
   $ sudo cp oc /usr/local/bin
   $ oc adm release extract --registry-config "$pullsecret_file" --command="$cmd" --to "$extract_dir" "$RELEASE_IMAGE"
   $ sudo cp openshift-baremetal-install /usr/local/bin
   ```

### 6.3.5. Creating an RHCOS images cache (optional)

To employ image caching, you must download two images: the Red Hat Enterprise Linux CoreOS (RHCOS) image used by the bootstrap VM and the RHCOS image used by the installer to provision the different nodes. Image caching is optional, but especially useful when running the installer on a network with limited bandwidth.

If you are running the installer on a network with limited bandwidth and the RHCOS images download takes more than 15 to 20 minutes, the installer will timeout. Caching images on a web server will help in such scenarios.

Use the following steps to install a container that contains the images.

1. Install podman.

   ```
   $ sudo dnf install -y podman
   ```
2. Open firewall port **8080** to be used for RHCOS image caching.

```
$ sudo firewall-cmd --add-port=8080/tcp --zone=public --permanent
$ sudo firewall-cmd --reload
```

3. Create a directory to store the **bootstraposimage** and **clusterosimage**.

```
$ mkdir /home/kni/rhcos_image_cache
```

4. Set the appropriate SELinux context for the newly created directory.

```
$ sudo semanage fcontext -a -t httpd_sys_content_t "/home/kni/rhcos_image_cache(/.*)?"
$ sudo restorecon -Rv rhcos_image_cache/
```

5. Get the commit ID from the installer. The ID determines which images the installer needs to download.

```
$ export COMMIT_ID=$(/usr/local/bin/openshift-baremetal-install version | grep "built from commit" | awk '{print $4}"
```

6. Get the URI for the RHCOS image that the installer will deploy on the nodes.

```
$ export RHCOS_OPENSTACK_URI=$(curl -s -S https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json | jq .images.openstack.path | sed 's/"//g')
```

7. Get the URI for the RHCOS image that the installer will deploy on the bootstrap VM.

```
$ export RHCOS_QEMU_URI=$(curl -s -S https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json | jq .images.qemu.path | sed 's/"//g')
```

8. Get the path where the images are published.

```
$ export RHCOS_PATH=$(curl -s -S https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json | jq .baseURI | sed 's/"//g')
```

9. Get the SHA hash for the RHCOS image that will be deployed on the bootstrap VM.

```
```

10. Get the SHA hash for the RHCOS image that will be deployed on the nodes.

```
$ export RHCOS_OPENSTACK_SHA_COMPRESSED=$(curl -s -S https://raw.githubusercontent.com/openshift/installer/$COMMIT_ID/data/data/rhcos.json | jq -r ".images.openstack.sha256")
```

11. Download the images and place them in the `/home/kni/rhcos_image_cache` directory.
$ curl -L ${RHCOS_PATH}${RHCOS_QEMU_URI} -o /home/kni/rhcos_image_cache/${RHCOS_QEMU_URI}  
$ curl -L ${RHCOS_PATH}${RHCOS_OPENSTACK_URI} -o /home/kni/rhcos_image_cache/${RHCOS_OPENSTACK_URI}

12. Confirm SELinux type is of httpd_sys_content_t for the newly created files.

   $ ls -Z /home/kni/rhcos_image_cache

13. Create the pod.

   $ podman run -d --name rhcos_image_cache \  
   -v /home/kni/rhcos_image_cache:/var/www/html \  
   -p 8080:8080/tcp \  
   registry.centos.org/centos/httpd-24-centos7:latest

### 6.3.6. Configuration files

#### 6.3.6.1. Configuring the install-config.yaml file

The **install-config.yaml** file requires some additional details. Most of the information is teaching the installer and the resulting cluster enough about the available hardware so that it is able to fully manage it.

1. Configure **install-config.yaml**. Change the appropriate variables to match the environment, including **pullSecret** and **sshKey**.

```yaml
apiVersion: v1
baseDomain: <domain>
metadata:
  name: <cluster-name>
networking:
  machineCIDR: <public-cidr>
networkType: OVNKubernetes
compute:
  - name: worker
    replicas: 2
controlPlane:
  name: master
  replicas: 3
platform:
  platform: baremetal: {}
platform:
  baremetal:
    apiVIP: <api-ip>
    ingressVIP: <wildcard-ip>
    provisioningNetworkInterface: <NIC1>
    provisioningNetworkCIDR: <CIDR>
  hosts:
    - name: openshift-master-0
      role: master
      bmc:
        address: ipmi://<out-of-band-ip>
```


1. Scale the worker machines based on the number of worker nodes that are part of the OpenShift Container Platform cluster.

2. Create a directory to store cluster configs.

   ```bash
   $ mkdir ~/clusterconfigs
   $ cp install-config.yaml ~/clusterconfigs
   ```

3. Ensure all bare metal nodes are powered off prior to installing the OpenShift Container Platform cluster.

   ```bash
   $ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
   ```

Refer to the BMC addressing sections for more options.
4. Remove old bootstrap resources if any are left over from a previous deployment attempt.

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk '{print $2'}); do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

6.3.6.2. Setting proxy settings within the install-config.yaml file (optional)

To deploy an OpenShift Container Platform cluster using a proxy, make the following changes to the install-config.yaml file.

```
apiVersion: v1
baseDomain: <domain>
proxy:
  httpProxy: http://USERNAME:PASSWORD@proxy.example.com:PORT
  httpsProxy: https://USERNAME:PASSWORD@proxy.example.com:PORT
noProxy: <WILDCARD_OF_DOMAIN>,<PROVISIONING_NETWORK/CIDR>,
        <BMC_ADDRESS_RANGE/CIDR>
```

The following is an example of noProxy with values.

```
noProxy: .example.com,172.22.0.0/24,10.10.0.0/24
```

With a proxy enabled, set the appropriate values of the proxy in the corresponding key/value pair.

Key considerations:

- If the proxy does not have an HTTPS proxy, change the value of httpsProxy from https:// to http://.
- If using a provisioning network, include it in the noProxy setting, otherwise the installer will fail.
- Set all of the proxy settings as environment variables within the provisioner node. For example, HTTP_PROXY, HTTPS_PROXY, and NO_PROXY.

6.3.6.3. Modifying the install-config.yaml file for no provisioning network (optional)

To deploy an OpenShift Container Platform cluster without a provisioning network, make the following changes to the install-config.yaml file.

```
platform:
  baremetal:
    apiVIP: <apiVIP>
    ingressVIP: <ingress/wildcard VIP>
    provisioningNetwork: "Disabled"
    provisioningHostIP: <baremetal_network_IP1>
    bootstrapProvisioningIP: <baremetal_network_IP2>
```
NOTE

Requires providing two IP addresses from the baremetal network for the provisioningHostIP and bootstrapProvisioningIP configuration settings, and removing the provisioningBridge and provisioningNetworkCIDR configuration settings.

6.3.6.4. Additional install-config parameters

See the following tables for the required parameters, the hosts parameter, and the bmc parameter for the install-config.yaml file.

Table 6.1. Required parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseDomain</td>
<td></td>
<td>The domain name for the cluster. For example, example.com.</td>
</tr>
<tr>
<td>sshKey</td>
<td></td>
<td>The sshKey configuration setting contains the key in the ~/.ssh/id_rsa.pub file required to access the control plane nodes and worker nodes. Typically, this key is from the provisioner node.</td>
</tr>
<tr>
<td>pullSecret</td>
<td></td>
<td>The pullSecret configuration setting contains a copy of the pull secret downloaded from the Install OpenShift on Bare Metal page when preparing the provisioner node.</td>
</tr>
<tr>
<td>metadata:</td>
<td></td>
<td>The name to be given to the OpenShift Container Platform cluster. For example, openshift.</td>
</tr>
<tr>
<td>name:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>networking:</td>
<td></td>
<td>The public CIDR (Classless Inter-Domain Routing) of the external network. For example, 10.0.0.0/24.</td>
</tr>
<tr>
<td>machineCIDR:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>compute:</td>
<td></td>
<td>The OpenShift Container Platform cluster requires a name be provided for worker (or compute) nodes even if there are zero nodes.</td>
</tr>
<tr>
<td>- name: worker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>replicas:</td>
<td></td>
<td>Replicas sets the number of worker (or compute) nodes in the OpenShift Container Platform cluster.</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The OpenShift Container Platform cluster requires a name for control plane (master) nodes.

Replicas sets the number of control plane (master) nodes included as part of the OpenShift Container Platform cluster.

The name of the network interface on control plane nodes connected to the provisioning network.

The default configuration used for machine pools without a platform configuration.

The VIP to use for internal API communication. This setting must either be provided or pre-configured in the DNS so that the default name resolves correctly.

redfish and redfish-virtualmedia need this parameter to manage BMC addresses. The value should be True when using a self-signed certificate for BMC addresses.

The VIP to use for ingress traffic.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane: name: master</td>
<td></td>
<td>The OpenShift Container Platform cluster requires a name for control plane (master) nodes.</td>
</tr>
<tr>
<td>controlPlane: replicas: 3</td>
<td></td>
<td>Replicas sets the number of control plane (master) nodes included as part of the OpenShift Container Platform cluster.</td>
</tr>
<tr>
<td>provisioningNetworkInterface</td>
<td></td>
<td>The name of the network interface on control plane nodes connected to the provisioning network.</td>
</tr>
<tr>
<td>defaultMachinePlatform</td>
<td></td>
<td>The default configuration used for machine pools without a platform configuration.</td>
</tr>
<tr>
<td>apiVIP</td>
<td>api.&lt;clusternameme.clusterdomain&gt;</td>
<td>The VIP to use for internal API communication. This setting must either be provided or pre-configured in the DNS so that the default name resolves correctly.</td>
</tr>
<tr>
<td>disableCertificateVerification</td>
<td>False</td>
<td>redfish and redfish-virtualmedia need this parameter to manage BMC addresses. The value should be True when using a self-signed certificate for BMC addresses.</td>
</tr>
<tr>
<td>ingressVIP</td>
<td>test.apps.&lt;clusternameme.clusterdomain&gt;</td>
<td>The VIP to use for ingress traffic.</td>
</tr>
</tbody>
</table>

Table 6.2. Optional Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provisioningDHCPRange</td>
<td>172.22.0.10,172.22.0.100</td>
<td>Defines the IP range for nodes on the provisioning network.</td>
</tr>
<tr>
<td>provisioningNetworkCIDR</td>
<td>172.22.0.0/24</td>
<td>The CIDR for the network to use for provisioning. This option is required when not using the default address range on the provisioning network.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>clusterProvisioningIP</td>
<td>The third IP address of the provisioningNetworkCIDR.</td>
<td>The IP address within the cluster where the provisioning services run. Defaults to the third IP address of the provisioning subnet. For example, <strong>172.22.0.3</strong>.</td>
</tr>
<tr>
<td>bootstrapProvisioningIP</td>
<td>The second IP address of the provisioningNetworkCIDR.</td>
<td>The IP on the bootstrap VM where the provisioning services run while the installer is deploying the control plane (master) nodes. Defaults to the second IP of the provisioning subnet. For example, <strong>172.22.0.2</strong>. When using no provisioning network, set this value to an IP address that is available on the baremetal network.</td>
</tr>
<tr>
<td>externalBridge</td>
<td>baremetal</td>
<td>The name of the baremetal bridge of the hypervisor attached to the baremetal network.</td>
</tr>
<tr>
<td>provisioningBridge</td>
<td>provisioning</td>
<td>The name of the provisioning bridge on the provisioner host attached to the provisioning network.</td>
</tr>
<tr>
<td>defaultMachine Platform</td>
<td></td>
<td>The default configuration used for machine pools without a platform configuration.</td>
</tr>
<tr>
<td>bootstrapOSImage</td>
<td></td>
<td>A URL to override the default operating system image for the bootstrap node. The URL must contain a SHA-256 hash of the image. For example: <strong><a href="https://mirror.openshift.com/rhcos-">https://mirror.openshift.com/rhcos-</a>&lt;version&gt;-qemu.qcow2.gz?sha256=&lt;uncompressed_sha256&gt;</strong>.</td>
</tr>
<tr>
<td>clusterOSImage</td>
<td></td>
<td>A URL to override the default operating system for cluster nodes. The URL must include a SHA-256 hash of the image. For example, <strong><a href="https://mirror.openshift.com/images/rhcos-">https://mirror.openshift.com/images/rhcos-</a>&lt;version&gt;-openstack.qcow2.gz?sha256=&lt;compressed_sha256&gt;</strong>.</td>
</tr>
<tr>
<td>provisioningNetwork</td>
<td></td>
<td>Set this parameter to <strong>Disabled</strong> to disable the requirement for a provisioning network. User may only do virtual media based provisioning, or bring up the cluster using assisted installation. If using power management, BMC’s must be accessible from the machine networks. User must provide two IP addresses on the external network that are used for the provisioning services. Set this parameter to <strong>managed</strong>, which is the default, to fully manage the provisioning network, including DHCP, TFTP, and so on. Set this parameter to <strong>unmanaged</strong> to still enable the provisioning network but take care of manual configuration of DHCP. Virtual Media provisioning is recommended but PXE is still available if required.</td>
</tr>
</tbody>
</table>
Hosts

The **hosts** parameter is a list of separate bare metal assets used to build the cluster.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>The name of the <strong>BareMetalHost</strong> resource to associate with the details. For example, <strong>openshift-master-0</strong>.</td>
</tr>
<tr>
<td>role</td>
<td></td>
<td>The role of the bare metal node. Either <strong>master</strong> or <strong>worker</strong>.</td>
</tr>
<tr>
<td>bmc</td>
<td></td>
<td>Connection details for the baseboard management controller. See the BMC addressing section for additional details.</td>
</tr>
<tr>
<td>bootMACAddress</td>
<td></td>
<td>The MAC address of the NIC the host will use to boot on the <strong>provisioning</strong> network.</td>
</tr>
</tbody>
</table>

6.3.6.5. BMC addressing

The **address** field for each **bmc** entry is a URL for connecting to the OpenShift Container Platform cluster nodes, including the type of controller in the URL scheme and its location on the network.

**IPMI**

IPMI hosts use `ipmi://<out-of-band-ip>:<port>` and defaults to port 623 if not specified. The following example demonstrates an IPMI configuration within the **install-config.yaml** file.
RedFish for HPE

To enable RedFish, use `redfish://` or `redfish+http://` to disable TLS. The installer requires both the hostname or the IP address and the path to the system ID. The following example demonstrates a RedFish configuration within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: ipmi://<out-of-band-ip>
          username: <user>
          password: <password>

platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/<id>
          username: <user>
          password: <password>

platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/<id>
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include `disableCertificateVerification: True` in the `bmc` configuration if using self-signed certificates. The following example demonstrates a RedFish configuration using the `disableCertificateVerification: True` configuration parameter within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/<id>
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

RedFish for Dell

To enable RedFish, use `redfish://` or `redfish+http://` to disable TLS. The installer requires both the hostname or the IP address and the path to the system ID. The following example demonstrates a RedFish configuration within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
```

```
1186
While it is recommended to have a certificate of authority for the out-of-band management addresses, you must include `disableCertificateVerification: True` in the `bmc` configuration if using self-signed certificates. The following example demonstrates a RedFish configuration using the `disableCertificateVerification: True` configuration parameter within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

**NOTE**

Currently RedFish is only supported on Dell with iDRAC firmware version 4.20.20.20 or higher for installer-provisioned installations of OpenShift Container Platform on bare metal deployments.

**RedFish Virtual Media for HPE**

To enable RedFish Virtual Media for HPE servers, use `redfish-virtualmedia://` in the `address` setting. The following example demonstrates using RedFish Virtual Media within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
```

**RedFish Virtual Media for Dell**

For RedFish Virtual Media on Dell servers, use `idrac-virtualmedia://` in the `address` setting.

**NOTE**

RedFish Virtual Media on Dell servers has a known issue in OpenShift Container Platform 4.6. The 4.6.1 point release will resolve the issue.

The following example demonstrates using iDRAC Virtual Media within the `install-config.yaml` file.

```yaml
platform:
  baremetal:
```
NOTE

idrac-virtualmedia requires iDRAC firmware version 4.20.20.20 or higher.

Ensure the OpenShift Container Platform cluster nodes have AutoAttach Enabled through the iDRAC console. The menu path is: Configuration→Virtual Media→Attach Mode→AutoAttach.

6.3.6.6. Root device hints

The rootDeviceHints parameter enables the installer to provision the Red Hat Enterprise Linux CoreOS (RHCOS) image to a particular device. The installer examines the devices in the order it discovers them, and compares the discovered values with the hint values. The installer uses the first discovered device that matches the hint value. The configuration can combine multiple hints, but a device must match all hints for the installer to select it.

Table 6.3. Subfields

<table>
<thead>
<tr>
<th>Subfield</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>deviceName</strong></td>
<td>A string containing a Linux device name like /dev/vda. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td><strong>hctl</strong></td>
<td>A string containing a SCSI bus address like 0:0:0:0. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td><strong>model</strong></td>
<td>A string containing a vendor-specific device identifier. The hint can be a substring of the actual value.</td>
</tr>
<tr>
<td><strong>vendor</strong></td>
<td>A string containing the name of the vendor or manufacturer of the device. The hint can be a substring of the actual value.</td>
</tr>
<tr>
<td><strong>serialNumber</strong></td>
<td>A string containing the device serial number. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td><strong>minSizeGigabytes</strong></td>
<td>An integer representing the minimum size of the device in gigabytes.</td>
</tr>
<tr>
<td><strong>wwn</strong></td>
<td>A string containing the unique storage identifier. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td>Subfield</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>wwnWithExtension</td>
<td>A string containing the unique storage identifier with the vendor extension appended. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td>wwnVendorExtension</td>
<td>A string containing the unique vendor storage identifier. The hint must match the actual value exactly.</td>
</tr>
<tr>
<td>rotational</td>
<td>A boolean indicating whether the device should be a rotating disk (true) or not (false).</td>
</tr>
</tbody>
</table>

**Example usage**

```yaml
- name: master-0
  role: master
  bmc:
    address: ipmi://10.10.0.3:6203
    username: admin
    password: redhat
    bootMACAddress: de:ad:be:ef:00:40
  rootDeviceHints:
    deviceName: '/dev/sda'
```

### 6.3.6.7. Creating the OpenShift Container Platform manifests

1. Create the OpenShift Container Platform manifests.

   ```bash
   $ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
   INFO Consuming Install Config from target directory
   WARNING Making control-plane schedulable by setting MastersSchedulable to true for Scheduler cluster settings
   WARNING Discarding the OpenShift Manifest that was provided in the target directory because its dependencies are dirty and it needs to be regenerated
   ```

### 6.3.7. Creating a disconnected registry (optional)

In some cases, you might want to install an OpenShift KNI cluster using a local copy of the installation registry. This could be for enhancing network efficiency because the cluster nodes are on a network that does not have access to the internet.

A local, or mirrored, copy of the registry requires the following:

- A certificate for the registry node. This can be a self-signed certificate.
- A webserver - this will be served by a container on a system.
- An updated pull secret that contains the certificate and local repository information.
NOTE

Creating a disconnected registry on a registry node is optional. The subsequent sections indicate that they are optional since they are steps you need to execute only when creating a disconnected registry on a registry node. You should execute all of the subsequent sub-sections labeled "(optional)" when creating a disconnected registry on a registry node.

6.3.7.1. Preparing the registry node to host the mirrored registry (optional)

Make the following changes to the registry node.

Procedure

1. Open the firewall port on the registry node.

   $ sudo firewall-cmd --add-port=5000/tcp --zone=libvirt --permanent
   $ sudo firewall-cmd --add-port=5000/tcp --zone=public --permanent
   $ sudo firewall-cmd --reload

2. Install the required packages for the registry node.

   $ sudo yum -y install python3 podman httpd httpd-tools jq

3. Create the directory structure where the repository information will be held.

   $ sudo mkdir -p /opt/registry/{auth,certs,data}

6.3.7.2. Generating the self-signed certificate (optional)

Generate a self-signed certificate for the registry node and put it in the /opt/registry/certs directory.

Procedure

1. Adjust the certificate information as appropriate.

   $ host_fqdn=$( hostname --long )
   $ cert_c="<Country Name>"   # Country Name (C, 2 letter code)
   $ cert_s="<State>"           # Certificate State (S)
   $ cert_l="<Locality>"        # Certificate Locality (L)
   $ cert_o="<Organization>"   # Certificate Organization (O)
   $ cert_ou="<Org Unit>"      # Certificate Organizational Unit (OU)
   $ cert_cn="${host_fqdn}"    # Certificate Common Name (CN)

   $ openssl req \
     -newkey rsa:4096 \
     -nodes \ 
     -sha256 \
     -keyout /opt/registry/certs/domain.key \ 
     -x509 \ 
     -days 365 \ 
     -out /opt/registry/certs/domain.crt \ 
     -addext "subjectAltName = DNS:${host_fqdn}" \
     -subj "/C=${cert_c}/ST=${cert_s}/L=${cert_l}/O=${cert_o}/OU=${cert_ou}/CN=${cert_cn}"
NOTE

When replacing `<Country Name>`, ensure that it only contains two letters. For example, `US`.

2. Update the registry node’s `ca-trust` with the new certificate.

```bash
$ sudo cp /opt/registry/certs/domain.crt /etc/pki/ca-trust/source/anchors/
$ sudo update-ca-trust extract
```

6.3.7.3. Creating the registry podman container (optional)

The registry container uses the `/opt/registry` directory for certificates, authentication files, and to store its data files.

The registry container uses `httpd` and needs an `htpasswd` file for authentication.

**Procedure**

1. Create an `htpasswd` file in `/opt/registry/auth` for the container to use.

   ```bash
   $ htpasswd -bBc /opt/registry/auth/htpasswd <user> <passwd>
   
   Replace `<user>` with the user name and `<passwd>` with the password.
   
2. Create and start the registry container.

   ```bash
   $ podman create \
   --name ocpdiscon-registry \
   -p 5000:5000 \
   -e "REGISTRY_AUTH=htpasswd" \
   -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry" \
   -e "REGISTRY_HTTP_SECRET=ALongRandomSecretForRegistry" \
   -e "REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd" \
   -e "REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt" \
   -e "REGISTRY_HTTP_TLS_KEY=/certs/domain.key" \
   -e "REGISTRY_COMPATIBILITY_SCHEMA1_ENABLED=true" \
   -v /opt/registry/data:/var/lib/registry:z \
   -v /opt/registry/auth:/auth:z \
   docker.io/library/registry:2
   
   $ podman start ocpdiscon-registry
   ```

6.3.7.4. Copy and update the pull-secret (optional)

Copy the pull secret file from the provisioner node to the registry node and modify it to include the authentication information for the new registry node.

**Procedure**

1. Copy the `pull-secret.txt` file.
2. Update the `host_fqdn` environment variable with the fully qualified domain name of the registry node.

```bash
$ host_fqdn=$( hostname --long )
```

3. Update the `b64auth` environment variable with the base64 encoding of the HTTP credentials used to create the `htpasswd` file.

```bash
$ b64auth=$( echo -n '<username>:<passwd>' | openssl base64 )
```

Replace `<username>` with the user name and `<passwd>` with the password.

4. Set the `AUTHSTRING` environment variable to use the base64 authorization string. The `$USER` variable is an environment variable containing the name of the current user.

```bash
$ AUTHSTRING="""$host_fqdn:5000": {"auth": "$b64auth","email": "$USER@redhat.com"}""
```

5. Update the `pull-secret.txt` file.

```bash
$ jq ".auths += $AUTHSTRING" < pull-secret.txt > pull-secret-update.txt
```

6.3.7.5. Mirroring the repository (optional)

**Procedure**

1. Copy the `oc` binary from the provisioner node to the registry node.

```bash
$ sudo scp kni@provisioner:/usr/local/bin/oc /usr/local/bin
```

2. Set the required environment variables.

   a. Set the release version:

   ```bash
   $ VERSION=<release_version>
   
   For `<release_version>`, specify the tag that corresponds to the version of OpenShift Container Platform to install, such as 4.6.
   
   b. Set the local registry name and host port:

   ```bash
   $ LOCAL_REG='<local_registry_host_name>:<local_registry_host_port>'
   
   For `<local_registry_host_name>`, specify the registry domain name for your mirror repository, and for `<local_registry_host_port>`, specify the port that it serves content on.
   
   c. Set the local repository name:

   ```bash
   $ LOCAL_REPO='<local_repository_name>'
   ```
3. Mirror the remote install images to the local repository.

```bash
$ /usr/local/bin/oc adm release mirror
   -a pull-secret-update.txt \
   --from=$UPSTREAM_REPO \
   --to-release-image=$LOCAL_REG/$LOCAL_REPO:${VERSION} \
   --to=$LOCAL_REG/$LOCAL_REPO
```

6.3.7.6. Modify the install-config.yaml file to use the disconnected registry (optional)

On the provisioner node, the `install-config.yaml` file should use the newly created pull-secret from the `pull-secret-update.txt` file. The `install-config.yaml` file must also contain the disconnected registry node’s certificate and registry information.

**Procedure**

1. Add the disconnected registry node’s certificate to the `install-config.yaml` file. The certificate should follow the "additionalTrustBundle: |" line and be properly indented, usually by two spaces.

   ```bash
   $ echo "additionalTrustBundle: |" >> install-config.yaml
   $ sed -e 's/^/  /' /opt/registry/certs/domain.crt >> install-config.yaml
   ```

2. Add the mirror information for the registry to the `install-config.yaml` file.

   ```bash
   $ echo "imageContentSources:" >> install-config.yaml
   $ echo "  - mirrors:" >> install-config.yaml
   $ echo "    - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
   $ echo "    source: quay.io/openshift-release-dev/ocp-release" >> install-config.yaml
   $ echo "  - mirrors:" >> install-config.yaml
   $ echo "    - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
   $ echo "    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev" >> install-config.yaml
   ```

**NOTE**

Replace `registry.example.com` with the registry’s fully qualified domain name.

6.3.8. Deploying routers on worker nodes

During installation, the installer deploys router pods on worker nodes. By default, the installer installs two router pods. If the initial cluster has only one worker node, or if a deployed cluster requires additional routers to handle external traffic loads destined for services within the OpenShift Container Platform cluster, you can create a `yaml` file to set an appropriate number of router replicas.

**NOTE**

By default, the installer deploys two routers. If the cluster has at least two worker nodes, you can skip this section. For more information on the Ingress Operator see: Ingress Operator in OpenShift Container Platform.
NOTE

If the cluster has no worker nodes, the installer deploys the two routers on the control plane nodes by default. If the cluster has no worker nodes, you can skip this section.

Procedure

1. Create a `router-replicas.yaml` file.

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: IngressController
   metadata:
     name: default
     namespace: openshift-ingress-operator
   spec:
     replicas: <num-of-router-pods>
     endpointPublishingStrategy:
       type: HostNetwork
     nodePlacement:
       nodeSelector:
         matchLabels:
           node-role.kubernetes.io/worker: ""
   
   NOTE

   Replace `<num-of-router-pods>` with an appropriate value. If working with just one worker node, set `replicas:` to 1. If working with more than 3 worker nodes, you can increase `replicas:` from the default value 2 as appropriate.

2. Save and copy the `router-replicas.yaml` file to the `clusterconfigs/openshift` directory.

   ```bash
   cp ~/router-replicas.yaml clusterconfigs/openshift/99_router-replicas.yaml
   ```

6.3.9. Validation checklist for installation

- OpenShift Container Platform installer has been retrieved.
- OpenShift Container Platform installer has been extracted.
- Required parameters for the `install-config.yaml` have been configured.
- The `hosts` parameter for the `install-config.yaml` has been configured.
- The `bmc` parameter for the `install-config.yaml` has been configured.
- Conventions for the values configured in the `bmc address` field have been applied.
- Created a disconnected registry (optional).
- (optional) Validate disconnected registry settings if in use.
- (optional) Deployed routers on worker nodes.

6.3.10. Deploying the cluster via the OpenShift Container Platform installer
Run the OpenShift Container Platform installer:

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

### 6.3.11. Following the installation

During the deployment process, you can check the installation’s overall status by issuing the `tail` command to the `.openshift_install.log` log file in the install directory folder.

```
$ tail -f /path/to/install-dir/.openshift_install.log
```

### 6.4. EXPANDING THE CLUSTER

After deploying an installer-provisioned OpenShift Container Platform cluster, you can use the following procedures to expand the number of worker nodes. Ensure that each prospective worker node meets the prerequisites.

#### 6.4.1. Preparing the bare metal node

Preparing the bare metal node requires executing the following procedure from the provisioner node.

**Procedure**

1. Get the `oc` binary, if needed. It should already exist on the provisioner node.
   
   ```
   $ curl -s https://mirror.openshift.com/pub/openshift-v4/clients/ocp-dev-preview/$VERSION/openshift-client-linux.tar.gz | tar zxvf -
   $ sudo cp oc /usr/local/bin
   ```

2. Install the `ipmitool`.

   ```
   $ sudo dnf install -y OpenIPMI ipmitool
   ```

3. Power off the bare metal node and ensure it is off.

   ```
   $ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
   
   Where `<management-server-ip>` is the IP address of the bare metal node’s base board management controller.
   ```

4. Retrieve the username and password of the bare metal node’s baseboard management controller. Then, create `base64` strings from the username and password. In the following example, the username is `root` and the password is `calvin`.

   ```
   $ echo -ne "root" | base64
   ```
$ echo -ne "calvin" | base64

5. Create a configuration file for the bare metal node.

$ vim bmh.yaml

---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret
type: Opaque
data:
  username: <base64-of-uid>
  password: <base64-of-pwd>
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num>
spec:
  online: true
  bootMACAddress: <NIC1-mac-address>
  bmc:
    address: ipmi://<bmc-ip>
    credentialsName: openshift-worker-<num>-bmc-secret

Replace <num> for the worker number of bare metal node in two name fields and credentialsName field. Replace <base64-of-uid> with the base64 string of the username. Replace <base64-of-pwd> with the base64 string of the password. Replace <NIC1-mac-address> with the MAC address of the bare metal node’s first NIC. Replace <bmc-ip> with the IP address of the bare metal node’s baseboard management controller.

6. Create the bare metal node.

$ oc -n openshift-machine-api create -f bmh.yaml

secret/openshift-worker-<num>-bmc-secret created
baremetalhost.metal3.io/openshift-worker-<num> created

Where <num> will be the worker number.

7. Power up and inspect the bare metal node.

$ oc -n openshift-machine-api get bmh openshift-worker-<num>

Where <num> is the worker node number.

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>PROVISIONING STATUS</th>
<th>CONSUMER</th>
<th>BMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-worker-&lt;num&gt;</td>
<td>OK</td>
<td>ready</td>
<td></td>
<td>ipmi://&lt;out-of-band-ip&gt;</td>
</tr>
</tbody>
</table>
6.4.2. Provisioning the bare metal node

Provisioning the bare metal node requires executing the following procedure from the provisioner node.

**Procedure**

1. Ensure the **PROVISIONING STATUS** is **ready** before provisioning the bare metal node.

   ```bash
   $ oc -n openshift-machine-api get bmh openshift-worker-<num>
   ``

   Where `<num>` is the worker node number.

2. Get a count of the number of worker nodes.

   ```bash
   $ oc get nodes
   ``

   ```text
   NAME                                                STATUS   ROLES           AGE     VERSION
   provisioner.openshift.example.com            Ready    master          30h     v1.16.2
   openshift-master-1.openshift.example.com            Ready    master          30h     v1.16.2
   openshift-master-2.openshift.example.com            Ready    master          30h     v1.16.2
   openshift-master-3.openshift.example.com            Ready    master          30h     v1.16.2
   openshift-worker-0.openshift.example.com            Ready    master          30h     v1.16.2
   openshift-worker-1.openshift.example.com            Ready    master          30h     v1.16.2
   ``

3. Get the machine set.

   ```bash
   $ oc get machinesets -n openshift-machine-api
   ``

   ```text
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   openshift-worker-0.example.com      1         1         1       1           55m
   openshift-worker-1.example.com      1         1         1       1           55m
   ``

4. Increase the number of worker nodes by one.

   ```bash
   $ oc scale --replicas=<num> machineset <machineset> -n openshift-machine-api
   ``

   Replace `<num>` with the new number of worker nodes. Replace `<machineset>` with the name of the machine set from the previous step.

5. Check the status of the bare metal node.

   ```bash
   $ oc -n openshift-machine-api get bmh openshift-worker-<num>
   ``

   Where `<num>` is the worker node number. The status changes from **ready** to **provisioning**.

   ```text
   NAME                 STATUS   PROVISIONING STATUS   CONSUMER   BMC
   openshift-worker-<num>   OK       ready                            ipmi://<out-of-band-ip>   unknown  true
   ```
The **provisioning** status remains until the OpenShift Container Platform cluster provisions the node. This can take 30 minutes or more. Once complete, the status will change to **provisioned**.

6. Once provisioned, ensure the bare metal node is ready.

```
$ oc get nodes
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>provisioner.openshift.example.com</td>
<td>Ready</td>
<td>master</td>
<td>30h</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-master-1.openshift.example.com</td>
<td>Ready</td>
<td>master</td>
<td>30h</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-master-2.openshift.example.com</td>
<td>Ready</td>
<td>master</td>
<td>30h</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-master-3.openshift.example.com</td>
<td>Ready</td>
<td>master</td>
<td>30h</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-worker-0.openshift.example.com</td>
<td>Ready</td>
<td>worker</td>
<td>3m27s</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-worker-1.openshift.example.com</td>
<td>Ready</td>
<td>worker</td>
<td>3m27s</td>
<td>v1.16.2</td>
</tr>
<tr>
<td>openshift-worker-&lt;num&gt;.openshift.example.com</td>
<td>Ready</td>
<td>worker</td>
<td>3m27s</td>
<td>v1.16.2</td>
</tr>
</tbody>
</table>

You can also check the kubelet.

```
$ ssh openshift-worker-<num>
```

```
[kni@openshift-worker-<num>]$ journalctl -fu kubelet
```

### 6.5. TROUBLESHOOTING

#### 6.5.1. Troubleshooting the installer workflow

Prior to troubleshooting the installation environment, it is critical to understand the overall flow of the installer-provisioned installation on bare metal. The diagrams below provide a troubleshooting flow with a step-by-step breakdown for the environment.
Workflow 1 of 4 illustrates a troubleshooting workflow when the `install-config.yaml` file has errors or the Red Hat Enterprise Linux CoreOS (RHCOS) images are inaccessible. Troubleshooting suggestions can be found at Troubleshooting `install-config.yaml`.

Workflow 2 of 4 illustrates a troubleshooting workflow for bootstrap VM issues, bootstrap VMs that cannot boot up the cluster nodes, and inspecting logs. When installing a OpenShift Container Platform cluster without the `provisioning` network, this workflow does not apply.
Workflow 3 of 4 illustrates a troubleshooting workflow for cluster nodes that will not PXE boot.

Workflow 4 of 4 illustrates a troubleshooting workflow from a non-accessible API to a validated installation.
6.5.2. Troubleshooting install-config.yaml

The **install-config.yaml** configuration file represents all of the nodes that are part of the OpenShift Container Platform cluster. The file contains the necessary options consisting of but not limited to `apiVersion`, `baseDomain`, `imageContentSources` and virtual IP addresses. If errors occur early in the deployment of the OpenShift Container Platform cluster, the errors are likely in the **install-config.yaml** configuration file.

**Procedure**

1. Use the guidelines in **YAML-tips**.
2. Verify the YAML syntax is correct using **syntax-check**.
3. Verify the Red Hat Enterprise Linux CoreOS (RHCOS) QEMU images are properly defined and accessible via the URL provided in the **install-config.yaml**. For example:

   ```bash
   $ curl -s -o /dev/null -I -w "%{http_code}\n" http://webserver.example.com:8080/rhcos-44.81.202004250133-0-qemu.x86_64.qcow2.gz?sha256=7d884b46ee54fe87bbc3893bf2aa99af3b2d31f2e19ab5529c60636fbdf0f1ce7
   If the output is **200**, there is a valid response from the webserver storing the bootstrap VM image.

6.5.3. Bootstrap VM issues

The OpenShift Container Platform installer spawns a bootstrap node virtual machine, which handles provisioning the OpenShift Container Platform cluster nodes.

**Procedure**

1. About 10 to 15 minutes after triggering the installer, check to ensure the bootstrap VM is operational using the **virsh** command:

   ```bash
   $ sudo virsh list
   Id    Name                           State
   --------------------------------------------
   12    openshift-xf6fq-bootstrap      running
   $ systemctl status libvirtd
   ● libvirtd.service - Virtualization daemon
   
   ![NOTE](image)

   The name of the bootstrap VM is always the cluster name followed by a random set of characters and ending in the word "bootstrap."

   If the bootstrap VM is not running after 10-15 minutes, troubleshoot why it is not running. Possible issues include:

2. Verify **libvirtd** is running on the system:

   ```bash
   $ systemctl status libvirtd
   • libvirtd.service - Virtualization daemon
If the bootstrap VM is operational, log in to it.

3. Use the `virsh console` command to find the IP address of the bootstrap VM:

   ![Code Snippet]

   ```
   $ sudo virsh console example.com
   ```

   Connected to domain example.com
   Escape character is ^J

   Red Hat Enterprise Linux CoreOS 43.81.202001142154.0 (Ootpa) 4.3
   SSH host key: SHA256:BRWJktXZgQQRY5zjuAV0lKZ4WM7i4TiUyMVanqu9Pqg (ED25519)
   SSH host key: SHA256:7+jKGA7VtG5szmk2jB5g/5EZ+SNCsJ3a2g23o0lnio (ECDSA)
   SSH host key: SHA256:DH5VWhhvagOTaLsYiVNse9ca+ZSW/30OOMed8rIGOc (RSA)
   ens4: 172.22.0.2 fe80::1d05:e52e:be5d:263f
   localhost login:

   ![Note]

   **IMPORTANT**

   When deploying a OpenShift Container Platform cluster without the `provisioning` network, you must use a public IP address and not a private IP address like `172.22.0.2`.

4. Once you obtain the IP address, log in to the bootstrap VM using the `ssh` command:

   ![Code Snippet]

   ```
   $ ssh core@172.22.0.2
   ```

   If you are not successful logging in to the bootstrap VM, you have likely encountered one of the following scenarios:

   - You cannot reach the `172.22.0.0/24` network. Verify network connectivity on the provisioner host specifically around the `provisioning` network bridge. This will not be the issue if you are not using the `provisioning` network.

   - You cannot reach the bootstrap VM via the public network. When attempting to SSH via `baremetal` network, verify connectivity on the `provisioner` host specifically around the `baremetal` network bridge.
You encountered **Permission denied (publickey,password,keyboard-interactive)**. When attempting to access the bootstrap VM, a **Permission denied** error might occur. Verify that the SSH key for the user attempting to log into the VM is set within the `install-config.yaml` file.

### 6.5.3.1. Bootstrap VM cannot boot up the cluster nodes

During the deployment, it is possible for the bootstrap VM to fail to boot the cluster nodes, which prevents the VM from provisioning the nodes with the RHCOS image. This scenario can arise due to:

- A problem with the `install-config.yaml` file.
- Issues with out-of-band network access via the baremetal network.

To verify the issue, there are three containers related to **ironic**:

- **ironic-api**
- **ironic-conductor**
- **ironic-inspector**

**Procedure**

1. Log in to the bootstrap VM:
   
   ```
   $ ssh core@172.22.0.2
   ```

2. To check the container logs, execute the following:

   ```
   [core@localhost ~]$ sudo podman logs -f <container-name>
   ```

   Replace `<container-name>` with one of **ironic-api**, **ironic-conductor**, or **ironic-inspector**. If you encounter an issue where the control plane nodes are not booting up via PXE, check the **ironic-conductor** pod. The **ironic-conductor** pod contains the most detail about the attempt to boot the cluster nodes, because it attempts to log in to the node over IPMI.

**Potential reason**

The cluster nodes might be in the **ON** state when deployment started.

**Solution**

Power off the OpenShift Container Platform cluster nodes before you begin the installation over IPMI:

```
$ ipmitool -I lanplus -U root -P <password> -H <out-of-band-ip> power off
```

### 6.5.3.2. Inspecting logs

When experiencing issues downloading or accessing the RHCOS images, first verify that the URL is correct in the `install-config.yaml` configuration file.

**Example of internal webserver hosting RHCOS images**

```
bootstrapOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-qemu.x86_64.qcow2.gz?
sha256=9d999f55ff1d4df7ed7c10658e5deec0d4c3c06095d34d36bf1cd127837e0c
```
The `ipa-downloader` and `coreos-downloader` containers download resources from a webserver or the external `quay.io` registry, whichever the `install-config.yaml` configuration file specifies. Verify the following two containers are up and running and inspect their logs as needed:

- `ipa-downloader`
- `coreos-downloader`

**Procedure**

1. Log in to the bootstrap VM:
   
   ```
   $ ssh core@172.22.0.2
   ```

2. Check the status of the `ipa-downloader` and `coreos-downloader` containers within the bootstrap VM:
   
   ```
   [core@localhost ~]$ sudo podman logs -f ipa-downloader
   [core@localhost ~]$ sudo podman logs -f coreos-downloader
   ```

   If the bootstrap VM cannot access the URL to the images, use the `curl` command to verify that the VM can access the images.

3. To inspect the `bootkube` logs that indicate if all the containers launched during the deployment phase, execute the following:
   
   ```
   [core@localhost ~]$ journalctl -xe
   [core@localhost ~]$ journalctl -b -f -u bootkube.service
   ```

4. Verify all the pods, including `dnsmasq`, `mariadb`, `httpd`, and `ironic`, are running:
   
   ```
   [core@localhost ~]$ sudo podman ps
   ```

5. If there are issues with the pods, check the logs of the containers with issues. To check the log of the `ironic-api`, execute the following:
   
   ```
   [core@localhost ~]$ sudo podman logs <ironic-api>
   ```

**6.5.4. Cluster nodes will not PXE boot**

When OpenShift Container Platform cluster nodes will not PXE boot, execute the following checks on the cluster nodes that will not PXE boot. This procedure does not apply when installing a OpenShift Container Platform cluster without the `provisioning` network.

**Procedure**

1. Check the network connectivity to the `provisioning` network.
2. Ensure PXE is enabled on the NIC for the **provisioning** network and PXE is disabled for all other NICs.

3. Verify that the `install-config.yaml` configuration file has the proper hardware profile and boot MAC address for the NIC connected to the **provisioning** network. For example:

```yaml
control plane node settings
```

```yaml
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
hardwareProfile: default
```

```yaml
Worker node settings
```

```yaml
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
hardwareProfile: unknown
```

### 6.5.5. The API is not accessible

When the cluster is running and clients cannot access the API, domain name resolution issues might impede access to the API.

**Procedure**

1. **Hostname Resolution**: Check the cluster nodes to ensure they have a fully qualified domain name, and not just `localhost.localdomain`. For example:

   ```bash
   $ hostname
   
   If a hostname is not set, set the correct hostname. For example:
   
   ```bash
   $ hostnamectl set-hostname <hostname>
   ```

2. **Incorrect Name Resolution**: Ensure that each node has the correct name resolution in the DNS server using `dig` and `nslookup`. For example:

   ```bash
   $ dig api.<cluster-name>.example.com
   ```
The output in the foregoing example indicates that the appropriate IP address for the api. <cluster-name>.example.com VIP is 10.19.13.86. This IP address should reside on the baremetal network.

6.5.6. Cleaning up previous installations

In the event of a previous failed deployment, remove the artifacts from the failed attempt before attempting to deploy OpenShift Container Platform again.

Procedure

1. Power off all bare metal nodes prior to installing the OpenShift Container Platform cluster:

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

2. Remove all old bootstrap resources if any are left over from a previous deployment attempt:

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk '{print $2}'); do
    sudo virsh destroy $i;
    sudo virsh undefine $i;
    sudo virsh vol-delete $i --pool $i;
    sudo virsh vol-delete $i.ign --pool $i;
    sudo virsh pool-destroy $i;
    sudo virsh pool-undefine $i;
done
```

3. Remove the following from the clusterconfigs directory to prevent Terraform from failing:

```
$ rm -rf ~/clusterconfigs/auth ~/clusterconfigs/terraform* ~/clusterconfigs/tls ~/clusterconfigs/metadata.json
```

6.5.7. Issues with creating the registry

When creating a disconnected registry, you might encounter a "User Not Authorized" error when attempting to mirror the registry. This error might occur if you fail to append the new authentication to the existing pull-secret.txt file.

Procedure

1. Check to ensure authentication is successful:

```
$ /usr/local/bin/oc adm release mirror
```
NOTE

Example output of the variables used to mirror the install images:

```
UPSTREAM_REPO=${RELEASE_IMAGE}
LOCAL_REG=<registry_FQDN>:<registry_port>
LOCAL_REPO='ocp4/openshift4'
```

The values of RELEASE_IMAGE and VERSION were set during the Retrieving OpenShift Installer step of the Setting up the environment for an OpenShift installation section.

2. After mirroring the registry, confirm that you can access it in your disconnected environment:

```
$ curl -k -u <user>:<password> https://registry.example.com:<registry-port>/v2/_catalog
{"repositories":["<Repo-Name>"]}
```

6.5.8. Miscellaneous issues

6.5.8.1. Addressing the runtime network not ready error

After the deployment of a cluster you might receive the following error:

```
```

The Cluster Network Operator is responsible for deploying the networking components in response to a special object created by the installer. It runs very early in the installation process, after the control plane (master) nodes have come up, but before the bootstrap control plane has been torn down. It can be indicative of more subtle installer issues, such as long delays in bringing up control plane (master) nodes or issues with apiserver communication.

Procedure

1. Inspect the pods in the openshift-network-operator namespace:

```
$ oc get all -n openshift-network-operator
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>pod/network-operator-69dfd7b577-bg89v</td>
<td>0/1</td>
<td>ContainerCreating</td>
<td>0</td>
<td>149m</td>
</tr>
</tbody>
</table>

2. On the provisioner node, determine that the network configuration exists:

```
$ kubectl get network.config.openshift.io cluster -oyaml
```

```
apiVersion: config.openshift.io/v1
```
If it does not exist, the installer did not create it. To determine why the installer did not create it, execute the following:

3. Check that the network-operator is running:

   $ kubectl -n openshift-network-operator get pods

4. Retrieve the logs:

   $ kubectl -n openshift-network-operator logs -l "name=network-operator"

On high availability clusters with three or more control plane (master) nodes, the Operator will perform leader election and all other Operators will sleep. For additional details, see Troubleshooting.

### 6.5.8.2. Cluster nodes not getting the correct IPv6 address over DHCP

If the cluster nodes are not getting the correct IPv6 address over DHCP, check the following:

1. Ensure the reserved IPv6 addresses reside outside the DHCP range.

2. In the IP address reservation on the DHCP server, ensure the reservation specifies the correct DHCP Unique Identifier (DUID). For example:

   # This is a dnsmasq dhcp reservation, 'id:00:03:00:01' is the client id and '18:db:f2:8c:d5:9f' is the MAC Address for the NIC
   id:00:03:00:01:18:db:f2:8c:d5:9f,openshift-master-1,[2620:52:0:1302::6]

3. Ensure that route announcements are working.

4. Ensure that the DHCP server is listening on the required interfaces serving the IP address ranges.

### 6.5.8.3. Cluster nodes not getting the correct hostname over DHCP

During IPv6 deployment, cluster nodes must get their hostname over DHCP. Sometimes the NetworkManager does not assign the hostname immediately. A control plane (master) node might report an error such as:
Failed Units: 2
NetworkManager-wait-online.service
nodeip-configuration.service

This error indicates that the cluster node likely booted without first receiving a hostname from the DHCP server, which causes kubelet to boot with a localhost.localdomain hostname. To address the error, force the node to renew the hostname.

Procedure

1. Retrieve the hostname:

   [core@master-X ~]$ hostname

   If the hostname is localhost, proceed with the following steps.

   **NOTE**

   Where X is the control plane node (also known as the master node) number.

2. Force the cluster node to renew the DHCP lease:

   [core@master-X ~]$ sudo nmcli con up "<bare-metal-nic>"

   Replace <bare-metal-nic> with the wired connection corresponding to the baremetal network.

3. Check hostname again:

   [core@master-X ~]$ hostname

4. If the hostname is still localhost.localdomain, restart NetworkManager:

   [core@master-X ~]$ sudo systemctl restart NetworkManager

5. If the hostname is still localhost.localdomain, wait a few minutes and check again. If the hostname remains localhost.localdomain, repeat the previous steps.

6. Restart the nodeip-configuration service:

   [core@master-X ~]$ sudo systemctl restart nodeip-configuration.service

   This service will reconfigure the kubelet service with the correct hostname references.

7. Reload the unit files definition since the kubelet changed in the previous step:

   [core@master-X ~]$ sudo systemctl daemon-reload

8. Restart the kubelet service:

   [core@master-X ~]$ sudo systemctl restart kubelet.service

9. Ensure kubelet booted with the correct hostname:
If the cluster node is not getting the correct hostname over DHCP after the cluster is up and running, such as during a reboot, the cluster will have a pending csr. Do not approve a csr, or other issues might arise.

Addressing a csr

1. Get CSRs on the cluster:

   $ oc get csr

2. Verify if a pending csr contains Subject Name: localhost.localdomain:

   $ oc get csr <pending_csr> -o jsonpath='{.spec.request}' | base64 --decode | openssl req -noout -text

3. Remove any csr that contains Subject Name: localhost.localdomain:

   $ oc delete csr <wrong_csr>

6.5.8.4. Routes do not reach endpoints

During the installation process, it is possible to encounter a Virtual Router Redundancy Protocol (VRRP) conflict. This conflict might occur if a previously used OpenShift Container Platform node that was once part of a cluster deployment using a specific cluster name is still running but not part of the current OpenShift Container Platform cluster deployment using that same cluster name. For example, a cluster was deployed using the cluster name openshift, deploying three control plane (master) nodes and three worker nodes. Later, a separate install uses the same cluster name openshift, but this redeployment only installed three control plane (master) nodes, leaving the three worker nodes from a previous deployment in an ON state. This might cause a Virtual Router Identifier (VRID) conflict and a VRRP conflict.

1. Get the route:

   $ oc get route oauth-openshift

2. Check the service endpoint:

   $ oc get svc oauth-openshift

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>oauth-openshift</td>
<td>ClusterIP</td>
<td>172.30.19.162</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>59m</td>
</tr>
</tbody>
</table>

3. Attempt to reach the service from a control plane (master) node:

   [core@master0 ~]$ curl -k https://172.30.19.162

```json
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {
```
4. Identify the **authentication-operator** errors from the **provisioner** node:

```bash
$ oc logs deployment/authentication-operator -n openshift-authentication-operator
```

Solution

1. Ensure that the cluster name for every deployment is unique, ensuring no conflict.

2. Turn off all the rogue nodes which are not part of the cluster deployment that are using the same cluster name. Otherwise, the authentication pod of the OpenShift Container Platform cluster might never start successfully.

### 6.5.8.5. Failed Ignition during Firstboot

During the Firstboot, the Ignition configuration may fail.

**Procedure**

1. Connect to the node where the Ignition configuration failed:

   Failed Units: 1
   machine-config-daemon-firstboot.service

2. Restart the **machine-config-daemon-firstboot** service:

   [core@worker-X ~]$ sudo systemctl restart machine-config-daemon-firstboot.service

### 6.5.8.6. NTP out of sync

The deployment of OpenShift Container Platform clusters depends on NTP synchronized clocks among the cluster nodes. Without synchronized clocks, the deployment may fail due to clock drift if the time difference is greater than two seconds.

**Procedure**

1. Check for differences in the **AGE** of the cluster nodes. For example:

   $ oc get nodes
2. Check for inconsistent timing delays due to clock drift. For example:

```bash
$ oc get bmh -n openshift-machine-api
master-1   error registering master-1  ipmi://<out-of-band-ip>

$ sudo timedatectl

Local time: Tue 2020-03-10 18:20:02 UTC
Universal time: Tue 2020-03-10 18:20:02 UTC
RTC time: Tue 2020-03-10 18:36:53
Time zone: UTC (UTC, +0000)
System clock synchronized: no
NTP service: active
RTC in local TZ: no
```

Addressing clock drift in existing clusters

1. Create a **chrony.conf** file and encode it as **base64** string. For example:

```bash
$ cat << EOF | base 64
server <NTP-server> iburst
stratumweight 0
driftfile /var/lib/chrony/drift
rtcsync
makectp 15
bindcmdaddress 127.0.0.1
bindc4address ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony
EOF
```

   **Replace** `<NTP-server>` **with the IP address of the NTP server. Copy the output.**

   ```text
   [text-in-base-64]
   ```

2. Create a **MachineConfig** object, replacing the **base64** string with the **[text-in-base-64]** string generated in the output of the previous step. The following example adds the file to the control plane (master) nodes. You can modify the file for worker nodes or make an additional machine config for the worker role.
$ cat << EOF > ./99_masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
creationTimestamp: null
labels:
machineconfiguration.openshift.io/role: master
name: 99-master-etc-chrony-conf
spec:
config:
ignition:
  config: {}
security:
tls: {}
timeouts: {}
version: 3.1.0
networkd: {}
passwd: {}
storage:
files:
  - contents:
      source: data:text/plain;charset=utf-8;base64,[text-in-base-64]
    group:
      name: root
      mode: 420
      overwrite: true
    path: /etc/chrony.conf
    user:
      name: root
    osImageURL: ""
EOF

1 Replace [text-in-base-64] with the base64 string.

3. Make a backup copy of the configuration file. For example:

   $ cp 99_masters-chrony-configuration.yaml 99_masters-chrony-configuration.yaml.backup

4. Apply the configuration file:

   $ oc apply -f ./masters-chrony-configuration.yaml

5. Ensure the System clock synchronized value is yes:

   $ sudo timedatectl

   Local time: Tue 2020-03-10 19:10:02 UTC
   Universal time: Tue 2020-03-10 19:10:02 UTC
   RTC time: Tue 2020-03-10 19:36:53
   Time zone: UTC (UTC, +0000)
   System clock synchronized: yes
   NTP service: active
   RTC in local TZ: no
To setup clock synchronization prior to deployment, generate the manifest files and add this file to the openshift directory. For example:

```
$ cp chrony-masters.yaml ~/clusterconfigs/openshift/99_masters-chrony-configuration.yaml
```

Then, continue to create the cluster.

### 6.5.9. Reviewing the installation

After installation, ensure the installer deployed the nodes and pods successfully.

**Procedure**

1. When the OpenShift Container Platform cluster nodes are installed appropriately, the following Ready state is seen within the STATUS column:

   ```
   $ oc get nodes
   NAME                   STATUS   ROLES           AGE  VERSION
   master-0.example.com   Ready    master,worker   4h   v1.16.2
   master-1.example.com   Ready    master,worker   4h   v1.16.2
   master-2.example.com   Ready    master,worker   4h   v1.16.2
   ```

2. Confirm the installer deployed all pods successfully. The following command removes any pods that are still running or have completed as part of the output.

   ```
   $ oc get pods --all-namespaces | grep -iv running | grep -iv complete
   ```
CHAPTER 7. INSTALLING ON IBM Z AND LINUXONE

7.1. INSTALLING A CLUSTER ON IBM Z AND LINUXONE

In OpenShift Container Platform version 4.6, you can install a cluster on IBM Z or LinuxONE infrastructure that you provision.

NOTE
While this document refers only to IBM Z, all information in it also applies to LinuxONE.

IMPORTANT
Additional considerations exist for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you install an OpenShift Container Platform cluster.

7.1.1. Prerequisites

- Before you begin the installation process, you must clean the installation directory. This ensures that the required installation files are created and updated during the installation process.

- Provision persistent storage using NFS for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE
Be sure to also review this site list if you are configuring a proxy.

7.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

7.1.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

7.1.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

NOTE

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

IMPORTANT

To improve high availability of your cluster, distribute the control plane machines over different z/VM instances on at least two physical machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

7.1.3.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config files from the Machine Config Server. The machines are configured with static IP addresses. No DHCP server is required. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server.

7.1.3.3. IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
• A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

### 7.1.3.4. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th>Machine</th>
<th>Operating System</th>
<th>vCPU</th>
<th>Virtual RAM</th>
<th>Storage</th>
<th>IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

### 7.1.3.5. Minimum IBM Z system environment

You can install OpenShift Container Platform version 4.6 on the following IBM hardware:

- IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- LinuxONE, any version

#### Hardware requirements

- The equivalent of 6 IFLs, which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the LoadBalancer service and to serve data for traffic outside the cluster.

**NOTE**

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.

**IMPORTANT**

Since the overall performance of the cluster can be impacted, the LPARs that are used to setup the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

#### Operating system requirements

- One instance of z/VM 7.1 or later

On your z/VM instance, set up:
- 3 guest virtual machines for OpenShift Container Platform control plane machines
- 2 guest virtual machines for OpenShift Container Platform compute machines
- 1 guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

**IBM Z network connectivity requirements**
To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:
- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

**Disk storage for the z/VM guest virtual machines**
- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV to ensure optimal performance.
- FCP attached disk storage

**Storage / Main Memory**
- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

**7.1.3.6. Preferred IBM Z system environment**

**Hardware requirements**
- 3 LPARS that each have the equivalent of 6 IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to connect to both connect to the LoadBalancer service and to serve data for traffic outside the cluster.
- HiperSockets, which are attached to a node either directly as a device or by bridging with one z/VM VSWITCH to be transparent to the z/VM guest. To directly connect HiperSockets to a node, you must set up a gateway to the external network via a RHEL 8 guest to bridge to the HiperSockets network.

**Operating system requirements**
- 2 or 3 instances of z/VM 7.1 or later for high availability

On your z/VM instances, set up:
- 3 guest virtual machines for OpenShift Container Platform control plane machines, one per z/VM instance.
- At least 6 guest virtual machines for OpenShift Container Platform compute machines, distributed across the z/VM instances.
- 1 guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using the CP command `SET SHARE`. Do the same for infrastructure nodes, if they exist. See `SET SHARE` in IBM Documentation.

**IBM Z network connectivity requirements**

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

**Disk storage for the z/VM guest virtual machines**

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV and High Performance FICON (zHPF) to ensure optimal performance.

- FCP attached disk storage

**Storage / Main Memory**

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

**7.1.3.7. Certificate signing requests management**

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

**Additional resources**

- See Bridging a HiperSockets LAN with a z/VM Virtual Switch in IBM Documentation.
- See Scaling HyperPAV alias devices on Linux guests on z/VM for performance optimization.
- See Topics in LPAR performance for LPAR weight management and entitlements.

**7.1.4. Creating the user-provisioned infrastructure**

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.
**Procedure**

1. Set up static IP addresses.
2. Set up an FTP server.
3. Provision the required load balancers.
4. Configure the ports for your machines.
5. Configure DNS.

**7.1.4.1. Networking requirements for user-provisioned infrastructure**

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require an FTP server in order to establish a network connection to download their Ignition config files.

Ensure that the machines have persistent IP addresses and host names.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

**Table 7.1. All machines to all machines**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
</tbody>
</table>
TCP/UDP | 30000-32767 | Kubernetes node port

Table 7.2. All machines to control plane

TCP | 6443 | Kubernetes API

Table 7.3. Control plane machines to control plane machines

TCP | 2379-2380 | etcd server and peer ports

Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**

Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

Table 7.4. API load balancer
<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

**Table 7.5. Application Ingress load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>
TIP
If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE
A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration
OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

Additional resources
- Configuring chrony time service

7.1.4.2. User-provisioned DNS requirements
DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 7.6. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Component</td>
<td>Record</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Routes</td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Master hosts</td>
<td>&lt;master&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Worker hosts</td>
<td>&lt;worker&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

---

**Example 7.1. Sample DNS zone database**

```plaintext
OpenShift Container Platform 4.6 Installing

1224
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 7.2. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (2019070700 ; serial
3H ; refresh (3 hours)
30M ; retry (30 minutes)
2W ; expiry (2 weeks)
1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
api.ocp4 IN A 192.168.1.5
api-int.ocp4 IN A 192.168.1.5
;
api.int.ocp4 IN A 192.168.1.5
;
*.apps.ocp4 IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4 IN A 192.168.1.97
master1.ocp4 IN A 192.168.1.98
master2.ocp4 IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4 IN A 192.168.1.11
worker1.ocp4 IN A 192.168.1.17
;
EOF
```
7.1.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

**IMPORTANT**

Do not skip this procedure in production environments where disaster recovery and debugging is required.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```

   1 Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.
NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   $ eval "$(ssh-agent -s)"

Example output

   Agent pid 31874

NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   $ ssh-add <path>/<file_name>

Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.1.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on your provisioning machine.

Prerequisites

- You have a machine that runs Linux, for example Red Hat Enterprise Linux 8, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your
operating system, and place the file in the directory where you will store the installation
configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to
install your cluster. You must keep the installation program and the files that the
installation program creates after you finish installing the cluster. Both files are
required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your
cluster, even if the cluster failed during installation. To remove your cluster,
complete the OpenShift Container Platform uninstallation procedures for your
specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating
system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull
secret allows you to authenticate with the services that are provided by the included authorities,
including Quay.io, which serves the container images for OpenShift Container Platform
components.

7.1.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a
command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands
in OpenShift Container Platform 4.6. Download and install the new version of oc.

7.1.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer
Portal.

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:
Place the `oc` binary in a directory that is on your `PATH`. To check your `PATH`, execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 7.1.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the `Version` drop-down menu.
3. Click `Download Now` next to the `OpenShift v4.6 Windows Client` entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the `oc` binary to a directory that is on your `PATH`. To check your `PATH`, open the command prompt and execute the following command:

```
C:\> path
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 7.1.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the `Version` drop-down menu.
3. Click `Download Now` next to the `OpenShift v4.6 MacOSX Client` entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your `PATH`. To check your `PATH`, open a terminal and execute the following command:

```
$ tar xvzf <file>
$ echo $PATH
$ oc <command>
```

C:\> path

C:\> oc <command>
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 7.1.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 7.1.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster's platform. When you create the `install-config.yaml` installation configuration file, you provide values for
the required parameters through the command line. If you customize your cluster, you can modify the
install-config.yaml file to provide more details about the platform.

NOTE

After installation, you cannot modify these parameters in the install-config.yaml file.

IMPORTANT

The openshift-install command does not validate field names for parameters. If an
incorrect name is specified, the related file or object is not created, and no error is
reported. Ensure that the field names for any parameters that are specified are correct.

### 7.1.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (-), such as dev.</td>
</tr>
</tbody>
</table>
platform

The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere.
For additional information about platform.<platform> parameters, consult the following table for your specific platform.

pullSecret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{ &quot;auths&quot;: { &quot;cloud.openshift.com&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;<a href="mailto:you@example.com">you@example.com</a>&quot; }, &quot;quay.io&quot;: { &quot;auth&quot;: &quot;b3Blb=&quot;, &quot;email&quot;: &quot;<a href="mailto:you@example.com">you@example.com</a>&quot; } } }</td>
</tr>
</tbody>
</table>

Note

You cannot modify parameters specified by the networking object after installation.

7.1.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 7.8. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

Note

You cannot modify parameters specified by the networking object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides $2^{32-23} - 2$ pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
</tbody>
</table>
networking.machineNetwork.cidr

Required if you use networking.machineNetwork. An IP address block. The default value is **10.0.0.0/16** for all platforms other than libvirt. For libvirt, the default value is **192.168.126.0/24**.

An IP network block in CIDR notation. For example, **10.0.0.0/16**.

**NOTE**

Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

7.1.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 7.9. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>compute.hyperthreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td><strong>compute.name</strong></td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><strong>compute.platform</strong></td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td><strong>compute.replicas</strong></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><strong>controlPlane</strong></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><strong>controlPlane.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

**controlPlane.hypertreading**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
</tbody>
</table>

**controlPlane.name**

Required if you use controlPlane. The name of the machine pool.

- master

**controlPlane.platform**

Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.

- aws, azure, gcp, openstack, ovirt, vsphere, or {}

**controlPlane.replicas**

The number of control plane machines to provision.

- The only supported value is 3, which is the default value.
## credentialsMode

The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

### NOTE

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is **false** (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**
The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**
If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>fips</strong></td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td><strong>imageContentSources</strong></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><strong>imageContentSources.source</strong></td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td><strong>imageContentSources.mirrors</strong></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>publish</strong></td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><strong>Internal</strong> or <strong>External</strong>. The default value is <strong>External</strong>. Setting this field to <strong>Internal</strong> is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td><strong>sshKey</strong></td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</td>
</tr>
</tbody>
</table>

### IMPORTANT

If the value of the field is set to **Internal**, the cluster will become non-functional. For more information, refer to [BZ#1953035](https://ibm.biz/BZ1953035).

### NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

#### 7.1.8.2. Sample install-config.yaml file for IBM Z

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
    architecture: s390x
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
  architecture: s390x
metadata:
  name: test
```
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    - cidr: 172.30.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
  fips: false
  pullSecret: '{"auths": ...}'}

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2. The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, -, and the first line of the **controlPlane** section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3. Whether to enable or disable simultaneous multithreading (SMT), or **hyperthreading**. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

   **NOTE**

   Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the **hyperthreading** parameter has no effect.

   **IMPORTANT**

   If you disable **hyperthreading**, whether in the BIOS or in the **install-config.yaml**, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4. You must set the value of the **replicas** parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

5. The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

6. The cluster name that you specified in your DNS records.

7. A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.
The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23, then each node is assigned a /23 subnet out of the given `cidr`, which allows for 510 \(2^{(32 - 23)} - 2\).

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to `none`. You cannot provide additional platform configuration variables for IBM Z infrastructure.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

### 7.1.9. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  
     noProxy: example.com  
   additionalTrustBundle: |  
     -----BEGIN CERTIFICATE-----  
     <MY_TRUSTED_CA_CERT>  
     -----END CERTIFICATE-----  
   ...
   ```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2. A proxy URL to use for creating HTTPS connections outside the cluster.
   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
   4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RH COS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RH COS trust bundle.

   NOTE

   The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster–wide proxy that is named **cluster** that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

### 7.1.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.
- You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   **Example output**

   ```bash
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   **For `<installation_directory>`**, specify the installation directory that contains the `install-config.yaml` file you created.
2. Check that the `mastersSchedulable` parameter in the 
<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
   a. Open the <installation_directory>/manifests/cluster-scheduler-02-config.yml file.
   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   c. Save and exit the file.

3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   $ ./openshift-install create ignition-configs --dir <installation_directory>  

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ├── auth
   │   └── kubeadmin-password
   │       └── kubeconfig
   │           └── bootstrap.ign
   │           └── master.ign
   │                   └── metadata.json
   │                           └── worker.ign

   7.1.11. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines

   Before you install a cluster on IBM Z infrastructure that you provision, you must install RHCOS on z/VM guest virtual machines for the cluster to use. Complete the following steps to create the machines.

   **Prerequisites**
   - An FTP server running on your provisioning machine that is accessible to the machines you create.

   **Procedure**

   1. Log in to Linux on your provisioning machine.

   2. Obtain the Red Hat Enterprise Linux CoreOS (RHCOS) kernel, initramfs, and rootfs files from the RHCOS image mirror.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate kernel, initramfs, and rootfs artifacts described in the following procedure.
The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel**: `rhcos-<version>-live-kernel-<architecture>`
- **initramfs**: `rhcos-<version>-live-initramfs.<architecture>.img`
- **rootfs**: `rhcos-<version>-live-rootfs.<architecture>.img`

**NOTE**

The rootfs image is the same for FCP and DASD.

3. Create parameter files. The following parameters are specific for a particular virtual machine:

- For `coreos.inst.install_dev=`, specify `dasda` for a DASD installation, or `sda` for FCP. Note that FCP requires `zfcp.allow_lun_scan=0`.
- For `rd.dasd=`, specify the DASD where RHCOS is to be installed.
- `rd.zfcp=<adapter>,<wwpn>,<lun>` specifies the FCP disk to install RHCOS on.
- For `ip=`, specify the following seven entries:
  i. The IP address for the machine.
  ii. An empty string.
  iii. The gateway.
  iv. The netmask.
  v. The machine host and domain name in the form `hostname.domainname`. Omit this value to let RHCOS decide.
  vi. The network interface name. Omit this value to let RHCOS decide.
  vii. If you use static IP addresses, an empty string.

- For `coreos.inst.ignition_url=`, specify the Ignition file for the machine role. Use `bootstrap.ign`, `master.ign`, or `worker.ign`. Only HTTP and HTTPS protocols are supported.
- For `coreos.live.rootfs_url=`, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.
- All other parameters can stay as they are.

Example parameter file, `bootstrap-0.parm`, for the bootstrap machine:

```plaintext
rd.neednet=1 \nconsole=ttySclp0 \ncoreos.inst.install_dev=dasda \ncoreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-roootfs.s390x.img \ncoreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \nip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 
```
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \ 
zfcp.allow_lun_scan=0 \ 
rd.dasd=0.0.3490

Write all options in the parameter file as a single line and make sure you have no newline characters.

4. Transfer the initramfs, kernel, parameter files, and RHCOS images to z/VM, for example with FTP. For details about how to transfer the files with FTP and boot from the virtual reader, see Installing under Z/VM.

5. Punch the files to the virtual reader of the z/VM guest virtual machine that is to become your bootstrap node.
   See PUNCH in IBM Documentation.

   **TIP**

   You can use the CP PUNCH command or, if you use Linux, the `vmur` command to transfer files between two z/VM guest virtual machines.


7. IPL the bootstrap machine from the reader:

   $ ipl c

   See IPL in IBM Documentation.

8. Repeat this procedure for the other machines in the cluster.

### 7.1.11.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

#### Routing and bonding options at RHCOS boot prompt

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

**IMPORTANT**

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.
## Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>To configure an IP address, either use DHCP (<code>ip=dhcp</code>) or set an individual static IP address (<code>ip=&lt;host_ip&gt;</code>). Then identify the DNS server IP address (<code>nameserver=&lt;dns_ip&gt;</code>) on each node. This example sets:</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41</code></td>
</tr>
<tr>
<td>• The node's IP address to <strong>10.10.10.2</strong></td>
<td></td>
</tr>
<tr>
<td>• The gateway address to <strong>10.10.10.254</strong></td>
<td></td>
</tr>
<tr>
<td>• The netmask to <strong>255.255.255.0</strong></td>
<td></td>
</tr>
<tr>
<td>• The hostname to <strong>core0.example.com</strong></td>
<td></td>
</tr>
<tr>
<td>• The DNS server address to <strong>4.4.4.41</strong></td>
<td></td>
</tr>
<tr>
<td>Specify multiple network interfaces by specifying multiple <code>ip=</code> entries.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code> <code>ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Optional: You can configure routes to additional networks by setting an <code>rd.route=</code> value. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.</td>
<td><code>ip=::10.10.10.254:::</code></td>
</tr>
<tr>
<td>To configure the default gateway:</td>
<td></td>
</tr>
<tr>
<td>To configure the route for the additional network:</td>
<td><code>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</code></td>
</tr>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=:::core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>You can combine DHCP and static IP configurations on systems with multiple network interfaces.</td>
<td><code>ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Optional: You can configure VLANs on individual interfaces by using the `vlan=` parameter. | To configure a VLAN on a network interface and use a static IP address:  
  ```  
  ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none  
  vlan=enp2s0.100:enp2s0  
  ```  
|                                                                           | To configure a VLAN on a network interface and to use DHCP:  
  ```  
  ip=enp2s0.100:dhcp  
  vlan=enp2s0.100:enp2s0  
  ```  
| You can provide multiple DNS servers by adding a `nameserver=` entry for each server. | nameserver=1.1.1.1  
|                                                                           | nameserver=8.8.8.8  
| Optional: Bonding multiple network interfaces to a single interface is supported using the `bond=` option. In these two examples: | To configure the bonded interface to use DHCP, set the bond’s IP address to `dhcp`. For example:  
  ```  
  bond=bond0:em1,em2:mode=active-backup  
  ip=bond0:dhcp  
  ```  
| • The syntax for configuring a bonded interface is:  
  ```  
  bond=name[:network_interfaces] [:options]  
  ```  
| • `name` is the bonding device name (bond0), `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (`em1,em2`), and `options` is a comma-separated list of bonding options. Enter `modinfo bonding` to see available options. | To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:  
  ```  
  bond=bond0:em1,em2:mode=active-backup  
  ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none  
  ```  
<p>| • When you create a bonded interface using <code>bond=</code>, you must specify how the IP address is assigned and other information for the bonded interface. | |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional: You can configure VLANs on bonded interfaces by using the <code>vlan=</code> parameter.</td>
<td>To configure the bonded interface with a VLAN and to use DHCP:</td>
</tr>
<tr>
<td></td>
<td>ip=bond0.100:dhcp</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup</td>
</tr>
<tr>
<td></td>
<td>vlan=bond0.100:bond0</td>
</tr>
<tr>
<td></td>
<td>To configure the bonded interface with a VLAN and to use a static IP address:</td>
</tr>
<tr>
<td></td>
<td>ip=10.10.10.2::10.10.254:255.255.255.255.0:core0.example.com:bond0.100:non</td>
</tr>
<tr>
<td></td>
<td>bond=bond0:em1,em2:mode=active-backup</td>
</tr>
<tr>
<td></td>
<td>vlan=bond0.100:bond0</td>
</tr>
<tr>
<td>Optional: Network teaming can be used as an alternative to bonding by using the <code>team=</code> parameter.</td>
<td>To configure a network team:</td>
</tr>
<tr>
<td>In this example:</td>
<td>team=team0:em1,em2</td>
</tr>
<tr>
<td></td>
<td>ip=team0:dhcp</td>
</tr>
<tr>
<td>![]</td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL.</td>
</tr>
<tr>
<td></td>
<td>For more information, see this <a href="https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/red_hat_enterprise_linux_system_administration/">Red Hat Knowledgebase Article</a>.</td>
</tr>
</tbody>
</table>

### 7.1.12. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**
1. Monitor the bootstrap process:

```bash
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \  
  --log-level=info
```

1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**Example output**

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

**IMPORTANT**

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

### 7.1.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```
7.1.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   
   **Example output**
   
   NAME    STATUS    ROLES   AGE   VERSION
   master-0 Ready  master  63m   v1.19.0
   master-1 Ready  master  63m   v1.19.0
   master-2 Ready  master  64m   v1.19.0
   
   The output lists all of the machines that you created.
   
   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```
   $ oc get csr
   
   **Example output**
   
   NAME        AGE     REQUESTOR                               CONDITION
   csr-mddf5   20m     system:node:master-01.example.com   Approved,Issued
   csr-z5rln   16m     system:node:worker-21.example.com   Approved,Issued
   
   3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:
NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```shell
  $ oc adm certificate approve <csr_name>  
  
  <csr_name> is the name of a CSR from the list of current CSRs.
  ```

- To approve all pending CSRs, run the following command:

  ```shell
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```shell
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:
  
  ```
  $ oc adm certificate approve <csr_name>  
  ```
  
  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:
  
  ```
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve  
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

  ```
  $ oc get nodes  
  ```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see **Certificate Signing Requests**.

7.1.15. **Initial Operator configuration**

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

  ```
  $ watch -n5 oc get clusteroperators  
  ```
2. Configure the Operators that are not available.

7.1.15.1. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

7.1.15.1.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**
Cluster administrator permissions.

A cluster on IBM Z.

Persistent storage provisioned for your cluster.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

Must have 100Gi capacity.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**
   
   If the storage type is **emptyDir**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```
   storage:
   pvc:
     claim:
   ```

   Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage** PVC.

4. Check the **clusteroperator** status:

   ```
   $ oc get clusteroperator image-registry
   ```

5. Ensure that your registry is set to managed to enable building and pushing of images.

   • Run:
$ oc edit configs.imageregistry/cluster

Then, change the line

managementState: Removed

to

managementState: Managed

7.1.15.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"storage":{"emptyDir":{}}}}'

  **WARNING**

  Configure this option for only non-production clusters.

  If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

  Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found

  Wait a few minutes and run the command again.

7.1.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.
- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```bash
$ watch -n5 oc get clusteroperators
```

### Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

      `$ oc get pods --all-namespaces`

      **Example output**

      | NAMESPACE                        | NAME                                           | READY | STATUS   |
      |---------------------------------|------------------------------------------------|-------|----------|
      | openshift-apiserver-operator    | openshift-apiserver-operator-85cb746d55-zqhs8  | 1/1   | Running  |
      |                                 |                                                 |       |          |
      |                                 | Running                                         | 1/1   |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver             | apiserver-67b9g                                | 1/1   | Running  |
      |                                 |                                                 |       |          |
      |                                 | Running                                         | 1/1   |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver             | apiserver-ljcmx                                | 1/1   | Running  |
      |                                 |                                                 |       |          |
      |                                 | Running                                         | 1/1   |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver             | apiserver-z25h4                                | 1/1   | Running  |
      |                                 |                                                 |       |          |
      |                                 | Running                                         | 1/1   |          |
      |                                 |                                                 |       |          |
      | openshift-authentication-operator| authentication-operator-69d5d8bf84-vh2n8        | 1/1   | Running  |
      |                                 |                                                 |       |          |
      |                                 | Running                                         | 1/1   |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver             |                                                 |       |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver-operator    |                                                 |       |          |
      |                                 |                                                 |       |          |
      | openshift-apiserver-operator    |                                                 |       |          |
      |                                 |                                                 |       |          |

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

      `$ oc logs <pod_name> -n <namespace>`

      1 Specify the pod name and namespace, as shown in the output of the previous command.

      If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

**7.1.17. Telemetry access for OpenShift Container Platform**
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

7.1.18. Collecting debugging information

You can gather debugging information that might help you to troubleshoot and debug certain issues with an OpenShift Container Platform installation on IBM Z.

Prerequisites

- The oc CLI tool installed.

Procedure

1. Log in to the cluster:

   

   $ oc login

2. On the node you want to gather hardware information about, start a debugging container:

   

   $ oc debug node/<nodename>

3. Change to the /host file system and start toolbox:

   

   $ chroot /host
   $ toolbox

4. Collect the dbginfo data:

   

   $ dbginfo.sh

5. You can then retrieve the data, for example, using scp.

Additional resources

- See How to generate SOSREPORT within OpenShift4 nodes without SSH.

7.1.19. Next steps

- Customize your cluster.

- If necessary, you can opt out of remote health reporting.
7.2. INSTALLING A CLUSTER ON IBM Z AND LINUXONE IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.6, you can install a cluster on IBM Z and LinuxONE infrastructure that you provision in a restricted network.

**NOTE**
While this document refers to only IBM Z, all information in it also applies to LinuxONE.

**IMPORTANT**
Additional considerations exist for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you install an OpenShift Container Platform cluster.

Prerequisites

- Create a mirror registry for installation in a restricted network and obtain the imageContentSources data for your version of OpenShift Container Platform.

- Before you begin the installation process, you must move or remove any existing installation files. This ensures that the required installation files are created and updated during the installation process.

  **IMPORTANT**
  Ensure that installation steps are done from a machine with access to the installation media.

- Provision persistent storage using NFS for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

  **NOTE**
  Be sure to also review this site list if you are configuring a proxy.

7.2.1. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.
To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

**IMPORTANT**

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

### 7.2.1.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an *Unable to retrieve available updates* error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

### 7.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access **Quay.io** to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 7.2.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

#### 7.2.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:
• One temporary bootstrap machine
• Three control plane, or master, machines
• At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To improve high availability of your cluster, distribute the control plane machines over different z/VM instances on at least two physical machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

7.2.3.2. **Network connectivity requirements**

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in **initramfs** during boot to fetch Ignition config files from the Machine Config Server. The machines are configured with static IP addresses. No DHCP server is required. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server.

7.2.3.3. **IBM Z network connectivity requirements**

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

7.2.3.4. **Minimum resource requirements**

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>N/A</td>
</tr>
</tbody>
</table>
1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: \((\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}\).

### 7.2.3.5. Minimum IBM Z system environment

You can install OpenShift Container Platform version 4.6 on the following IBM hardware:

- IBM z15 (all models), IBM z14 (all models), IBM z13, and IBM z13s
- LinuxONE, any version

**Hardware requirements**

- The equivalent of 6 IFLs, which are SMT2 enabled, for each cluster.
- At least one network connection to both connect to the **LoadBalancer** service and to serve data for traffic outside the cluster.

**NOTE**

You can use dedicated or shared IFLs to assign sufficient compute resources. Resource sharing is one of the key strengths of IBM Z. However, you must adjust capacity correctly on each hypervisor layer and ensure sufficient resources for every OpenShift Container Platform cluster.

**IMPORTANT**

Since the overall performance of the cluster can be impacted, the LPARs that are used to setup the OpenShift Container Platform clusters must provide sufficient compute capacity. In this context, LPAR weight management, entitlements, and CPU shares on the hypervisor level play an important role.

**Operating system requirements**

- One instance of z/VM 7.1 or later

On your z/VM instance, set up:

- 3 guest virtual machines for OpenShift Container Platform control plane machines
- 2 guest virtual machines for OpenShift Container Platform compute machines
- 1 guest virtual machine for the temporary OpenShift Container Platform bootstrap machine

**IBM Z network connectivity requirements**

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

**Disk storage for the z/VM guest virtual machines**

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the
minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you need extended address volumes (EAV). If available, use HyperPAV to ensure optimal performance.

- FCP attached disk storage

### Storage / Main Memory

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

### 7.2.3.6. Preferred IBM Z system environment

#### Hardware requirements

- 3 LPARs that each have the equivalent of 6 IFLs, which are SMT2 enabled, for each cluster.
- Two network connections to connect to both connect to the LoadBalancer service and to serve data for traffic outside the cluster.
- HiperSockets, which are attached to a node either directly as a device or by bridging with one z/VM VSwitch to be transparent to the z/VM guest. To directly connect HiperSockets to a node, you must set up a gateway to the external network via a RHEL 8 guest to bridge to the HiperSockets network.

#### Operating system requirements

- 2 or 3 instances of z/VM 7.1 or later for high availability

On your z/VM instances, set up:

- 3 guest virtual machines for OpenShift Container Platform control plane machines, one per z/VM instance.
- At least 6 guest virtual machines for OpenShift Container Platform compute machines, distributed across the z/VM instances.
- 1 guest virtual machine for the temporary OpenShift Container Platform bootstrap machine.
- To ensure the availability of integral components in an overcommitted environment, increase the priority of the control plane by using the CP command `SET SHARE`. Do the same for infrastructure nodes, if they exist. See `SET SHARE` in IBM Documentation.

#### IBM Z network connectivity requirements

To install on IBM Z under z/VM, you require a single z/VM virtual NIC in layer 2 mode. You also need:

- A direct-attached OSA or RoCE network adapter
- A z/VM VSwitch set up. For a preferred setup, use OSA link aggregation.

#### Disk storage for the z/VM guest virtual machines

- FICON attached disk storage (DASDs). These can be z/VM minidisks, fullpack minidisks, or dedicated DASDs, all of which must be formatted as CDL, which is the default. To reach the minimum required DASD size for Red Hat Enterprise Linux CoreOS (RHCOS) installations, you
need extended address volumes (EAV). If available, use HyperPAV and High Performance FICON (zHPF) to ensure optimal performance.

- FCP attached disk storage

**Storage / Main Memory**

- 16 GB for OpenShift Container Platform control plane machines
- 8 GB for OpenShift Container Platform compute machines
- 16 GB for the temporary OpenShift Container Platform bootstrap machine

### 7.2.3.7. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

**Additional resources**

- See [Bridging a HiperSockets LAN with a z/VM Virtual Switch](#) in IBM Documentation.
- See [Scaling HyperPAV alias devices on Linux guests on z/VM](#) for performance optimization.
- See [Topics in LPAR performance](#) for LPAR weight management and entitlements.

### 7.2.4. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

### 7.2.4.1. Networking requirements for user-provisioned infrastructure
All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in **initramfs** during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

### Table 7.10. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

### Table 7.11. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

### Table 7.12. Control plane machines to control plane machines
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

**IMPORTANT**
Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Table 7.13. API load balancer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>6443</td>
</tr>
<tr>
<td>22623</td>
</tr>
</tbody>
</table>
NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.
   - A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**Additional resources**
7.2.4.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 7.15. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

Example 7.3. Sample DNS zone database

```bash
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4 IN A 192.168.1.5
api-int.ocp4 IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4 IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4 IN A 192.168.1.97
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

```bind
$TTL 1W
@ IN SOA ns1.example.com. root (2019070700 ; serial
3H ; refresh (3 hours)
30M ; retry (30 minutes)
2W ; expiry (2 weeks)
1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

### Example 7.4. Sample DNS zone database for reverse records

7.2.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.
IMPORTANT

Do not skip this procedure in production environments where disaster recovery and debugging is required.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "\n-f <path>/<file_name>  
```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```
$ eval "$(ssh-agent -s)"
```

   Example output

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name>  
```

   Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

7.2.6. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.
- Obtain the `imageContentSources` section from the output of the command to mirror the repository.
- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```bash
   $ mkdir <installation_directory>
   ``

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

   - Unless you use a registry that RHCOS trusts by default, such as `docker.io`, you must provide the contents of the certificate for your mirror repository in the `additionalTrustBundle` section. In most cases, you must provide the certificate for your mirror.

   - You must include the `imageContentSources` section from the output of the command to mirror the repository.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
7.2.6.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

NOTE

After installation, you cannot modify these parameters in the `install-config.yaml` file.

IMPORTANT

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

7.2.6.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.16. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <strong>v1</strong>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <strong>ObjectMeta</strong>, from which only the <strong>name</strong> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}. {{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about <strong>platform.&lt;platform&gt;</strong> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{  &quot;auths&quot;:{  &quot;cloud.openshift.com&quot;:{  &quot;auth&quot;:&quot;b3Blb=&quot;,  &quot;email&quot;:&quot;you@example.com&quot;  },  &quot;quay.io&quot;:{  &quot;auth&quot;:&quot;b3Blb=&quot;,  &quot;email&quot;:&quot;you@example.com&quot;  }  }  }</code></td>
</tr>
</tbody>
</table>

### 7.2.6.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 7.17. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP kernel arguments, the machineNetwork.cidr value must be the CIDR of the primary network.</td>
<td>machineNetwork:</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>NOTE Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
<td></td>
</tr>
</tbody>
</table>

7.2.6.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 7.18. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**Internal** or **External**. The default value is **External**.

Setting this field to **Internal** is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to **Internal**, the cluster will become non-functional. For more information, refer to [BZ#1953035](#).

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your [ssh-agent](#) process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 7.2.6.2. Sample install-config.yaml file for IBM Z

You can customize the *install-config.yaml* file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com 1
compute: 2
  - hyperthreading: Enabled 3
    name: worker
    replicas: 0 4
    architecture: s390x
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
  architecture: s390x
metadata:
  name: test 8
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not.

Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

NOTE

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the hyperthreading parameter has no effect.

IMPORTANT

If you disable hyperthreading, whether in the BIOS or in the install-config.yaml, ensure that your capacity planning accounts for the dramatically decreased machine performance.

You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.
The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control

The cluster name that you specified in your DNS records.

A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23, then each node is assigned a /23 subnet out of the given cidr, which allows for 510 ($2^{32} - 2^{23} - 2$) pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to none. You cannot provide additional platform configuration variables for IBM Z infrastructure.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

For <local_registry>, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example registry.example.com or registry.example.com:5000. For <credentials>, specify the base64-encoded user name and password for your mirror registry.

The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

Add the additionalTrustBundle parameter and value. The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

Provide the imageContentSources section from the output of the command to mirror the repository.
7.2.6.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
     noProxy: example.com  # 3
     additionalTrustBundle: |
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
       ...
   ...
   ```

   **1** A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   
   **2** A proxy URL to use for creating HTTPS connections outside the cluster.
   
   **3** A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
   
   **4** If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to...
reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

7.2.7. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.

- You created the `install-config.yaml` installation configuration file.

**Procedure**
1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

```bash
$ ./openshift-install create manifests --dir <installation_directory>
```

**Example output**

INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift

For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```bash
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

**7.2.8. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines**

Before you install a cluster on IBM Z infrastructure that you provision, you must install RHCOS on z/VM guest virtual machines for the cluster to use. Complete the following steps to create the machines.

**Prerequisites**

- An FTP server running on your provisioning machine that is accessible to the machines you create.
Procedure

1. Log in to Linux on your provisioning machine.

2. Obtain the Red Hat Enterprise Linux CoreOS (RHCOS) kernel, initramfs, and rootfs files from the RHCOS image mirror.

   IMPORTANT

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate kernel, initramfs, and rootfs artifacts described in the following procedure.

   The file names contain the OpenShift Container Platform version number. They resemble the following examples:

   - kernel: rhcos-<version>-live-kernel-<architecture>
   - initramfs: rhcos-<version>-live-initramfs.<architecture>.img
   - rootfs: rhcos-<version>-live-rootfs.<architecture>.img

   NOTE

   The rootfs image is the same for FCP and DASD.

3. Create parameter files. The following parameters are specific for a particular virtual machine:

   - For `coreos.inst.install_dev=`, specify `dasda` for a DASD installation, or `sda` for FCP. Note that FCP requires `zfcp.allow_lun_scan=0`.
   - For `rd.dasd=`, specify the DASD where RHCOS is to be installed.
   - `rd.zfcp=<adapter>,<wwpn>,<lun>` specifies the FCP disk to install RHCOS on.
   - For `ip=`, specify the following seven entries:
     i. The IP address for the machine.
     ii. An empty string.
     iii. The gateway.
     iv. The netmask.
     v. The machine host and domain name in the form `hostname.domainname`. Omit this value to let RHCOS decide.
     vi. The network interface name. Omit this value to let RHCOS decide.
     vii. If you use static IP addresses, an empty string.
   - For `coreos.inst.ignition_url=`, specify the Ignition file for the machine role. Use `bootstrap.ign`, `master.ign`, or `worker.ign`. Only HTTP and HTTPS protocols are supported.
For coreos.live.rootfs_url=, specify the matching rootfs artifact for the kernel and initramfs you are booting. Only HTTP and HTTPS protocols are supported.

All other parameters can stay as they are.

Example parameter file, bootstrap-0.parm, for the bootstrap machine:

```
rd.neednet=1 \
coreos.live.rootfs_url=http://cl1.provide.example.com:8080/assets/rhcos-live-rootfs.s390x.img \
coreos.inst.ignition_url=http://cl1.provide.example.com:8080/ignition/bootstrap.ign \
lp=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rznet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
```

Write all options in the parameter file as a single line and make sure you have no newline characters.

4. Transfer the initramfs, kernel, parameter files, and RHCOS images to z/VM, for example with FTP. For details about how to transfer the files with FTP and boot from the virtual reader, see Installing under Z/VM.

5. Punch the files to the virtual reader of the z/VM guest virtual machine that is to become your bootstrap node.

   See PUNCH in IBM Documentation.

   **TIP**

   You can use the CP PUNCH command or, if you use Linux, the vmur command to transfer files between two z/VM guest virtual machines.


7. IPL the bootstrap machine from the reader:

   ```
   $ ipl c
   ```

   See IPL in IBM Documentation.

8. Repeat this procedure for the other machines in the cluster.

### 7.2.8.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

**Routing and bonding options at RHCOS boot prompt**

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.
IMPORTANT

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

NOTE

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| To configure an IP address, either use DHCP (`ip=dhcp`) or set an individual static IP address (`ip=<host_ip>`). Then identify the DNS server IP address (`nameserver=<dns_ip>`) on each node. This example sets: | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none`  
  * The node’s IP address to `10.10.10.2`  
  * The gateway address to `10.10.10.254`  
  * The netmask to `255.255.255.0`  
  * The hostname to `core0.example.com`  
  * The DNS server address to `4.4.4.41` |
| Specify multiple network interfaces by specifying multiple `ip=` entries. | `ip=10.10.10.2::10.10.254:255.255.255.0:core0.example.com:enp1s0:none`  
`ip=10.10.10.3::10.10.254:255.255.255.0:core0.example.com:enp2s0:none` |
| Optional: You can configure routes to additional networks by setting an `rd.route=` value. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway. | To configure the default gateway:  
`ip=:10.10.10.254:::`  
To configure the route for the additional network:  
`rd.route=20.20.20.0/24:20.20.20.254:enp2s0` |
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used. | `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none`  
`ip=:::core0.example.com:enp2s0:none`  

You can combine DHCP and static IP configurations on systems with multiple network interfaces. | `ip=enp1s0:dhcp`  
`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`  

Optional: You can configure VLANs on individual interfaces by using the `vlan=` parameter. | To configure a VLAN on a network interface and use a static IP address:  
`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none`  
`vlan=enp2s0.100:enp2s0`  
To configure a VLAN on a network interface and to use DHCP:  
`ip=enp2s0.100:dhcp`  
`vlan=enp2s0.100:enp2s0`  

You can provide multiple DNS servers by adding a `nameserver=` entry for each server. | `nameserver=1.1.1.1`  
`nameserver=8.8.8.8`  

Optional: Bonding multiple network interfaces to a single interface is supported using the `bond=` option. In these two examples:  
- The syntax for configuring a bonded interface is:  
  `bond=name[::network_interfaces] [::options]`  
- `name` is the bonding device name (bond0),  
  `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (em1,em2), and `options` is a comma-separated list of bonding options. Enter `modinfo bonding` to see available options.  
- When you create a bonded interface using `bond=`, you must specify how the IP address is assigned and other information for the bonded interface. | To configure the bonded interface to use DHCP, set the bond’s IP address to `dhcp`. For example:  
`bond=bond0:em1,em2:mode=active-backup`  
`ip=bond0:dhcp`  
To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:  
`bond=bond0:em1,em2:mode=active-backup`  
`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional: You can configure VLANs on bonded interfaces by using the <code>vlan=</code> parameter.</td>
<td>To configure the bonded interface with a VLAN and to use DHCP:</td>
</tr>
</tbody>
</table>
| | `ip=bond0.100:dhcp`  
`bond=bond0:em1,em2:mode=active-backup`  
`vlan=bond0.100:bond0` |
| | To configure the bonded interface with a VLAN and to use a static IP address: |
| | `ip=10.10.10.2::10.10.254:255.255.255.0:core0.example.com:bond0.100:None`  
`bond=bond0:em1,em2:mode=active-backup`  
`vlan=bond0.100:bond0` |
| Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example: | To configure a network team: |
| * The syntax for configuring a team interface is: `team=name[:network_interfaces]` where `name` is the team device name (team0) and `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (em1, em2). | `team=team0:em1,em2`  
`ip=team0:dhcp` |

**NOTE**

Teaming is planned to be deprecated when RHCS switches to an upcoming version of RHEL. For more information, see this [Red Hat Knowledgebase Article](#).  

### 7.2.9. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCS machines for your cluster.
Procedure

1. Monitor the bootstrap process:

   ```sh
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   
   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   **IMPORTANT**
   
   You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

7.2.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   ```sh
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:
7.2.11. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   $ oc get nodes

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   $ oc get csr

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

   **NOTE**

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.

   **NOTE**

   For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

   - To approve them individually, run the following command for each valid CSR:
     ```
     $ oc adm certificate approve <csr_name>
     ```

     `<csr_name>` is the name of a CSR from the list of current CSRs.

   - To approve all pending CSRs, run the following command:
     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
     ```

     **NOTE**

     Some Operators might not become available until some CSRs are approved.

   4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

     ```
     $ oc get csr
     ```

     **Example output**
5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name> 1
  
  1 <csr_name> is the name of a CSR from the list of current CSRs.
  
- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```bash
$ oc get nodes
```

### Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>Pending</td>
<td></td>
<td>5m26s</td>
<td></td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>Pending</td>
<td></td>
<td>5m26s</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Additional information

- For more information on CSRs, see [Certificate Signing Requests](#).

### 7.2.12. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

### Prerequisites

- Your control plane has initialized.
Procedure

1. Watch the cluster components come online:

   ```
   $ watch -n5 oc get clusteroperators
   ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h19m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

7.2.12.1. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:
Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

7.2.12.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

7.2.12.2.1. Configuring registry storage for IBM Z

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on IBM Z.
- Persistent storage provisioned for your cluster.

IMPORTANT

OpenShift Container Platform supports ReadWriteOnce access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, ReadWriteMany access is required.

- Must have 100Gi capacity.

Procedure

1. To configure your registry to use storage, change the spec.storage.pvc in the configs.imageregistry/cluster resource.

NOTE

When using shared storage, review your security settings to prevent outside access.

$ oc patch OperatorHub cluster --type json \  
   -p "["op": "add", "path": "/spec/disableAllDefaultSources", "value": true]"
2. Verify that you do not have a registry pod:

   $ oc get pod -n openshift-image-registry

   **NOTE**

   If the storage type is `emptyDIR`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   $ oc edit configs.imageregistry.operator.openshift.io

   **Example output**

   ```
   storage:
   pvc:
     claim:
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   $ oc get clusteroperator image-registry

5. Ensure that your registry is set to managed to enable building and pushing of images.

   - Run:

     ```
     $ oc edit configs.imageregistry/cluster
     ```

     Then, change the line

     ```
     managementState: Removed
     ```

     to

     ```
     managementState: Managed
     ```

7.2.12.2.2 Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

  ```
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"storage":{"emptyDir":{}}}'}
  ```
If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

### 7.2.13. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINCE authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```
   $ oc get pods --all-namespaces
   ```

   **Example output**
b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   $ oc logs <pod_name> -n <namespace>  

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Register your cluster on the Cluster registration page.

7.2.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

7.2.15. Collecting debugging information

You can gather debugging information that might help you to troubleshoot and debug certain issues with an OpenShift Container Platform installation on IBM Z.

Prerequisites

- The oc CLI tool installed.

Procedure

1. Log in to the cluster:
$ oc login

2. On the node you want to gather hardware information about, start a debugging container:

$ oc debug node/<nodename>

3. Change to the /host file system and start toolbox:

$ chroot /host
$ toolbox

4. Collect the dbginfo data:

$ dbginfo.sh

5. You can then retrieve the data, for example, using scp.

Additional resources

- See How to generate SOSREPORT within OpenShift Container Platform version 4 nodes without SSH.

7.2.16. Next steps

- Customize your cluster.

- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
CHAPTER 8. INSTALLING ON IBM POWER SYSTEMS

8.1. INSTALLING A CLUSTER ON IBM POWER SYSTEMS

In OpenShift Container Platform version 4.6, you can install a cluster on IBM Power Systems infrastructure that you provision.

IMPORTANT

Additional considerations exist for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you install an OpenShift Container Platform cluster.

Prerequisites

- Before you begin the installation process, you must clean the installation directory. This ensures that the required installation files are created and updated during the installation process.

- Provision persistent storage using NFS for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

8.1.1. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.
8.1.2. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

8.1.2.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

8.1.2.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

8.1.2.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>2</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>2</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>
### Machine Specifications

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
<td>RHCOS</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

#### 8.1.2.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

#### 8.1.3. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.

2. Provision the required load balancers.

3. Configure the ports for your machines.

4. Configure DNS.

5. Ensure network connectivity.

#### 8.1.3.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config from the machine config server.
During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

### Table 8.1. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

### Table 8.2. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

### Table 8.3. Control plane machines to control plane machines
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**

   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 8.4. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

Table 8.5. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

TIP

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources
8.1.3.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 8.6. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMPORTANT</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>
## Table of Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Record Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master hosts</td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Worker hosts</td>
<td>Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

### TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

### Example 8.1. Sample DNS zone database

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial
3H ; refresh (3 hours)
30M ; retry (30 minutes)
2W ; expiry (2 weeks)
1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.

ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5

api.ocp4 IN A 192.168.1.5
api-int.ocp4 IN A 192.168.1.5

*.apps.ocp4 IN A 192.168.1.5

bootstrap.ocp4 IN A 192.168.1.96

master0.ocp4 IN A 192.168.1.97
master1.ocp4 IN A 192.168.1.98
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

```bash
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

8.1.4. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.
NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\\
   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```text
   Agent pid 31874
   ```

   NOTE
   
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name> 1
   ```

   **Example output**

   ```text
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

8.1.5. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

8.1.6. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

8.1.6.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

$ oc <command>

8.1.6.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
8.1.6.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.

To check your PATH, open a terminal and execute the following command:

```bash
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the oc command:

```bash
$ oc <command>
```

8.1.7. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

**IMPORTANT**

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.
2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

**NOTE**
You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**
The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 8.1.7.1. Sample `install-config.yaml` file for IBM Power Systems

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
    architecture: ppc64le
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
  architecture: ppc64le
metadata:
  name: test
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2-5 The `controlPlane` section is a single mapping, but the `compute` section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not.
Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

**NOTE**

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the hyperthreading parameter has no effect.

**IMPORTANT**

If you disable hyperthreading, whether in the BIOS or in the `install-config.yaml`, ensure that your capacity planning accounts for the dramatically decreased machine performance.

You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.

A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23, then each node is assigned a /23 subnet out of the given cidr, which allows for 510 \(2^{32} - 2\) - 2) pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to none. You cannot provide additional platform configuration variables for IBM Power Systems infrastructure.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.
IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

8.1.7.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the install-config.yaml file.

Prerequisites

- You have an existing install-config.yaml file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform ( RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with `.` to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.

If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

**8.1.8. Creating the Kubernetes manifest and Ignition config files**

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.

- You created the `install-config.yaml` installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Example output

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```
1. For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

### 8.1.9. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines

Before you install a cluster on IBM Power Systems infrastructure that you provision, you must create RHCOS machines for it to use. Follow either the steps to use an ISO image or network PXE booting to create the machines.

#### 8.1.9.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on IBM Power Systems infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.

**Procedure**

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**

   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.
ISO file names resemble the following example:

```
rhcos-<version>-live.<architecture>.iso
```

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:
   - Burn the ISO image to a disk and boot it directly.
   - Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.

5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the `coreos-installer`.

6. Run the `coreos-installer` command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

   ```
   $ sudo coreos-installer install \
   --ignition-url=https://host/worker.ign /dev/sda
   ```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.

   **IMPORTANT**

   You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

8.1.9.1.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

**Routing and bonding options at RHCOS boot prompt**

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

   **IMPORTANT**

   When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.
NOTE

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>To configure an IP address, either use DHCP (<code>ip=dhcp</code>) or set an individual static IP address (<code>ip= &lt;host_ip&gt;</code>). Then identify the DNS server IP address (<code>nameserver=&lt;dns_ip&gt;</code>) on each node. This example sets:</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code>&lt;br&gt;<code>nameserver=4.4.4.41</code></td>
</tr>
<tr>
<td>• The node's IP address to 10.10.10.2</td>
<td></td>
</tr>
<tr>
<td>• The gateway address to 10.10.10.254</td>
<td></td>
</tr>
<tr>
<td>• The netmask to 255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>• The hostname to core0.example.com</td>
<td></td>
</tr>
<tr>
<td>• The DNS server address to 4.4.4.41</td>
<td></td>
</tr>
<tr>
<td>Specify multiple network interfaces by specifying multiple <code>ip=</code> entries.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code>&lt;br&gt;<code>ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Optional: You can configure routes to additional networks by setting an <code>rd.route=</code> value. If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.</td>
<td><code>ip=:10.10.10.254:::</code>&lt;br&gt;<code>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</code></td>
</tr>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code>&lt;br&gt;<code>ip=:::core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| You can combine DHCP and static IP configurations on systems with multiple network interfaces. | ip=enp1s0:dhcp  
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none |
| Optional: You can configure VLANs on individual interfaces by using the `vlan=` parameter. | To configure a VLAN on a network interface and use a static IP address:  
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none  
`vlan=enp2s0.100:enp2s0`  
To configure a VLAN on a network interface and to use DHCP:  
ip=enp2s0.100:dhcp  
`vlan=enp2s0.100:enp2s0` |
| You can provide multiple DNS servers by adding a `nameserver=` entry for each server. | `nameserver=1.1.1.1`  
`nameserver=8.8.8.8` |
| Optional: Bonding multiple network interfaces to a single interface is supported using the `bond=` option. In these two examples: | To configure the bonded interface to use DHCP, set the bond’s IP address to `dhcp`. For example:  
`bond=bond0:em1,em2:mode=active-backup`  
`ip=bond0:dhcp`  
To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:  
`bond=bond0:em1,em2:mode=active-backup`  
`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none` |
| - The syntax for configuring a bonded interface is:  
`bond=name[:network_interfaces] [:options]`  
- `name` is the bonding device name (bond0), `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (em1,em2), and `options` is a comma-separated list of bonding options. Enter `modinfo bonding` to see available options.  
- When you create a bonded interface using `bond=`, you must specify how the IP address is assigned and other information for the bonded interface. | |
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional: You can configure VLANs on bonded interfaces by using the <code>vlan=</code> parameter.</td>
<td>To configure the bonded interface with a VLAN and to use DHCP:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>ip=bond0.100:dhcp</code></td>
</tr>
<tr>
<td></td>
<td><code>bond=bond0:em1,em2:mode=active-backup</code></td>
</tr>
<tr>
<td></td>
<td><code>vlan=bond0.100:bond0</code></td>
</tr>
<tr>
<td>To configure the bonded interface with a VLAN and to use a static IP address:</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:0</code></td>
</tr>
<tr>
<td></td>
<td><code>bond=bond0:em1,em2:mode=active-backup</code></td>
</tr>
<tr>
<td></td>
<td><code>vlan=bond0.100:bond0</code></td>
</tr>
<tr>
<td>Optional: Network teaming can be used as an alternative to bonding by using the <code>team=</code> parameter. In this example:</td>
<td>To configure a network team:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>team=team0:em1,em2</code></td>
</tr>
<tr>
<td></td>
<td><code>ip=team0:dhcp</code></td>
</tr>
<tr>
<td>NOTES</td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td>Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this Red Hat Knowledgebase Article.</td>
</tr>
</tbody>
</table>

### 8.1.9.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Configure suitable PXE or iPXE infrastructure.
Have access to an HTTP server that you can access from your computer.

Procedure

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**
   
   You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS **kernel**, **initramfs** and **rootfs** files from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

   The file names contain the OpenShift Container Platform version number. They resemble the following examples:

   - **kernel**: `rhcos-<version>-live-kernel-<architecture>`
   - **initramfs**: `rhcos-<version>-live-initramfs.<architecture>.img`
   - **rootfs**: `rhcos-<version>-live-rootfs.<architecture>.img`

3. Upload the additional files that are required for your booting method:

   - For traditional PXE, upload the **kernel** and **initramfs** files to your TFTP server and the **rootfs** file to your HTTP server.
   - For iPXE, upload the **kernel**, **initramfs**, and **rootfs** files to your HTTP server.

   **IMPORTANT**
   
   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images.

   Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

   - For PXE:
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture>
  <architecture>.img coreos.live.roots_url=http://<HTTP_server>/rhcos-<version>-live-
  rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign

1 Specify the location of the live kernel file that you uploaded to your HTTP server. The
  URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

2 If you use multiple NICs, specify a single interface in the ip option. For example, to use
  DHCP on a NIC that is named eno1, set ip=eno1:dhcp.

3 Specify locations of the RHCOS files that you uploaded to your HTTP server. The
  initrd parameter value is the location of the initramfs file, the coreos.live.roots_url
  parameter value is the location of the roots file, and the coreos.inst.ignition_url
  parameter value is the location of the bootstrap Ignition config file. You can also add
  more kernel arguments to the APPEND line to configure networking or other boot
  options.

NOTE

This configuration does not enable serial console access on machines with a
graphical console. To configure a different console, add one or more
console= arguments to the APPEND line. For example, add console=tty0
console=ttyS0 to set the first PC serial port as the primary console and the
graphical console as a secondary console. For more information, see How
does one set up a serial terminal and/or console in Red Hat Enterprise
Linux?

- For iPXE:

  kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
  <architecture>.img coreos.inst.install_dev=/dev/sda
  coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
  <architecture>.img

1 Specify locations of the RHCOS files that you uploaded to your HTTP server. The
  kernel parameter value is the location of the kernel file, the initrd=main argument is
  needed for booting on UEFI systems, the coreos.live.roots_url parameter value is
  the location of the roots file, and the coreos.inst.ignition_url parameter value is the
  location of the bootstrap Ignition config file.

2 If you use multiple NICs, specify a single interface in the ip option. For example, to use
  DHCP on a NIC that is named eno1, set ip=eno1:dhcp.

3 Specify the location of the initramfs file that you uploaded to your HTTP server.
NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `kernel` line. For example, add `console=tty0 console=ttyS0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?

6. If you use PXE UEFI, perform the following actions:
   a. Provide the `shimx64.efi` and `grubx64.efi` EFI binaries and the `grub.cfg` file that are required for booting the system.
      - Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the `images/efiboot.img` file to your host:

         ```
         $ mkdir -p /mnt/iso
         $ mkdir -p /mnt/efiboot
         $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
         $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
         $$ mkdir -p /mnt/iso
         $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
         $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
         $ umount /mnt/efiboot
         $ umount /mnt/iso
         ```
      - From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

         ```
         $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
         $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
         $ umount /mnt/efiboot
         ```
   b. Edit the `grub.cfg` file to include arguments similar to the following:

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
      linuxefi rhcos-<version>-live-kernel<architecture> coreos.inst.install_dev=/dev/sda
      coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
      initrd rhcos-<version>-live-initramfs.<architecture>.img
      }
      ```

      where:
rhcos-<version>-live-kernel-<architecture>
Specifies the kernel file that you uploaded to your TFTP server.

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
Specifies the location of the live rootfs image that you uploaded to your HTTP server.

http://<HTTP_server>/bootstrap.ign
Specifies the location of the bootstrap Ignition config file that you uploaded to your HTTP server.

rhcos-<version>-live-initramfs.<architecture>.img
Specifies the location of the initramfs file that you uploaded to your TFTP server.

**NOTE**

7. Continue to create the machines for your cluster.

**IMPORTANT**
You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

### 8.1.10. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**

1. Monitor the bootstrap process:

   ```bash
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete
   --log-level=info
   ``

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

---

1330
Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

8.1.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami

   Example output

   system:admin

8.1.12. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.
Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  63m  v1.19.0
   master-1  Ready     master  63m  v1.19.0
   master-2  Ready     master  64m  v1.19.0
   ```

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```
   $ oc get csr
   ```

   **Example output**

   ```
   NAME        AGE     REQUESTOR                                                                 CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
   csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper Pending
   ...
   ```

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:
CHAPTER 8. INSTALLING ON IBM POWER SYSTEMS

NOTE
Because the CSRs rotate automatically, approve your CSRs within an hour of
adding the machines to the cluster. If you do not approve them within an hour, the
certificates will rotate, and more than two certificates will be present for each
node. You must approve all of these certificates. Once the client CSR is
approved, the Kubelet creates a secondary CSR for the serving certificate, which
requires manual approval. Then, subsequent serving certificate renewal requests
are automatically approved by the machine-approver if the Kubelet requests a
new certificate with identical parameters.

NOTE
For clusters running on platforms that are not machine API enabled, such as bare
metal and other user-provisioned infrastructure, you must implement a method
of automatically approving the kubelet serving certificate requests (CSRs). If a
request is not approved, then the oc exec, oc rsh, and oc logs commands
cannot succeed, because a serving certificate is required when the API server
connects to the kubelet. Any operation that contacts the Kubelet endpoint
requires this certificate approval to be in place. The method must watch for new
CSRs, confirm that the CSR was submitted by the node-bootstrapper service
account in the system:node or system:admin groups, and confirm the identity
of the node.
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

To approve all pending CSRs, run the following command:
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve

NOTE
Some Operators might not become available until some CSRs are approved.
4. Now that your client requests are approved, you must review the server requests for each
machine that you added to the cluster:
$ oc get csr

Example output
NAME
AGE REQUESTOR
CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...

1333


5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name>
  ```

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}\" | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the Ready status. Verify this by running the following command:

  ```bash
  $ oc get nodes
  ```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the Ready status.

**Additional information**

- For more information on CSRs, see Certificate Signing Requests.

### 8.1.13. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

  ```bash
  $ watch -n5 oc get clusteroperators
  ```
Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

### 8.1.13.1. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

#### 8.1.13.1.1. Configuring registry storage for IBM Power Systems

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**
- Cluster administrator permissions.
- A cluster on IBM Power Systems.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   
   **NOTE**

   If the storage type is **emptyDir**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   
   **Example output**

   ```yaml
   storage:
   pvc:
   claim:
   
   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```bash
   $ oc get clusteroperator image-registry
   
   **Example output**

   ```yaml
   
5. Ensure that your registry is set to managed to enable building and pushing of images.

   - Run:
8.13.1.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

Procedure

- To set the image registry storage to an empty directory:

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{
    "storage":{
      "emptyDir":[]}}}'
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

8.1.14. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```bash
$ watch -n5 oc get clusteroperators
```

### Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```bash
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

### Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.
### IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```bash
   $ oc get pods --all-namespaces
   ```

   **Example output**

   ```
   NAMESPACE                         NAME                                            READY   STATUS
   RESTARTS AGE
   openshift-apiserver-operator     openshift-apiserver-operator-85cb746d55-zqh8s     1/1     Running     1          9m
   openshift-apiserver              apiserver-67b9g                                 1/1     Running     0
   3m                                openshift-apiserver              apiserver-ljcmx                                 1/1     Running     0
   1m                                openshift-apiserver              apiserver-z25h4                                 1/1     Running     2m
   2m                                openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8     1/1     Running     0
   5m                                ...
   ```

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```bash
   $ oc logs <pod_name> -n <namespace>
   ```

   **Specify the pod name and namespace, as shown in the output of the previous command.**

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

### 8.1.15. Telemetry access for OpenShift Container Platform
In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

8.1.16. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.

8.2. INSTALLING A CLUSTER ON IBM POWER SYSTEMS IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.6, you can install a cluster on IBM Power Systems infrastructure that you provision in a restricted network.

**IMPORTANT**

Additional considerations exist for non-bare metal platforms. Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you install an OpenShift Container Platform cluster.

Prerequisites

- Create a mirror registry for installation in a restricted network and obtain the imageContentSources data for your version of OpenShift Container Platform.
- Before you begin the installation process, you must move or remove any existing installation files. This ensures that the required installation files are created and updated during the installation process.

**IMPORTANT**

Ensure that installation steps are performed on a machine with access to the installation media.

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.
8.2.1. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

8.2.1.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.

- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

8.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](https://openshift.io) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

8.2.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

8.2.3.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

NOTE

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

8.2.3.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

8.2.3.3. Minimum resource requirements
Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>2</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>2</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: \((\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}\).

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

8.2.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

8.2.4. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

Prerequisites

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.

Procedure

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.
8.2.4.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 8.7. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td>TCP</td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td>TCP</td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td>TCP</td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td>UDP</td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td>UDP</td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 8.8. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>
Table 8.9. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

Table 8.10. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. Application Ingress load balancer: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

Table 8.11. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

TIP

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

Additional resources
8.2.4.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 8.12. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes</td>
<td>api..&lt;cluster_name&gt;..&lt;base_domain&gt;</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int..&lt;cluster_name&gt;..&lt;base_domain&gt;</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps..&lt;cluster_name&gt;..&lt;base_domain&gt;</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap..&lt;cluster_name&gt;..&lt;base_domain&gt;</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>
Master hosts  <master><n>.<cluster_name>.<base_domain>.  DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts  <worker><n>.<cluster_name>.<base_domain>.  Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 8.3. Sample DNS zone database**

```bash
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4 IN A 192.168.1.5
api-int.ocp4 IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4 IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4 IN A 192.168.1.97
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 8.4. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

8.2.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.
NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874

   NOTE
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```
   $ ssh-add <path>/<file_name>
   ```

   Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

8.2.6. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

- Obtain the `imageContentSources` section from the output of the command to mirror the repository.

- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

   - Unless you use a registry that RHCOS trusts by default, such as `docker.io`, you must provide the contents of the certificate for your mirror repository in the `additionalTrustBundle` section. In most cases, you must provide the certificate for your mirror.

   - You must include the `imageContentSources` section from the output of the command to mirror the repository.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
IMPORTANT

The install-config.yaml file is consumed during the next step of the installation process. You must back it up now.

8.2.6.1. Sample install-config.yaml file for IBM Power Systems

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
deviceVendor: IBM
baseDomain: example.com
compute:
  - name: worker
    replicas: 0
    architecture: ppc64le
controlPlane:
  name: master
  replicas: 3
  architecture: ppc64le
metadata:
  name: test

networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
  hostPrefix: 23

platform:
  source:
    quay.io/openshift-release-dev/ocp-release

imageContentSources:
  - mirrors:
      - source: quay.io/openshift-release-dev/ocp-release

---
# Base domain
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

# Control Plane
The control plane settings.

# Compute
The compute settings.

---
# Platform
The platform settings.

---
# Networking
The networking settings.

---
# Additional Trust Bundle
The additional trust bundle.

---
# Image Content Sources
The image content sources.
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.
The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, - , and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading (SMT), or hyperthreading. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

NOTE
Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the hyperthreading parameter has no effect.

IMPORTANT
If you disable hyperthreading, whether in the BIOS or in the install-config.yaml, ensure that your capacity planning accounts for the dramatically decreased machine performance.

4 You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

7 The number of control plane machines that you add to the cluster. Because the cluster uses this value as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

8 The cluster name that you specified in your DNS records.

9 A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

10 The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23, then each node is assigned a /23 subnet out of the given cidr, which allows for $510 (2^(32 - 23) - 2)$ pod IPs addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

11 The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

12 You must set the platform to none. You cannot provide additional platform configuration variables for IBM Power Systems infrastructure.

13 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.
IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

Provide the contents of the certificate file that you used for your mirror registry.

Provide the `imageContentSources` section from the output of the command to mirror the repository.

8.2.6.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure
1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>  # A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
  httpsProxy: https://<username>:<pswd>@<ip>:<port>  # A proxy URL to use for creating HTTPS connections outside the cluster.
  noProxy: example.com  # A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
additionalTrustBundle: |  # If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
  ...
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
2. A proxy URL to use for creating HTTPS connections outside the cluster.
3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.
4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

**NOTE**

Only the Proxy object named cluster is supported, and no additional proxies can be created.

### 8.2.7. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.
The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.
- You created the **install-config.yaml** installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```sh
   $ ./openshift-install create manifests --dir <installation_directory>  
   ```

   **Example output**

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the **install-config.yaml** file you created.

2. Check that the **mastersSchedulable** parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   b. Locate the **mastersSchedulable** parameter and ensure that it is set to `false`.
   c. Save and exit the file.
3. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

```
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

For `<installation_directory>`, specify the same installation directory.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

### 8.2.8. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines

Before you install a cluster on IBM Power Systems infrastructure that you provision, you must create RHCOS machines for it to use. Follow either the steps to use an ISO image or network PXE booting to create the machines.

#### 8.2.8.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on IBM Power Systems infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.

**Procedure**

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**

   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the RHCOS image mirror page.
IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

ISO file names resemble the following example:

rhcos-<version>-live.<architecture>.iso

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:
   - Burn the ISO image to a disk and boot it directly.
   - Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.

5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the `coreos-installer`.

6. Run the `coreos-installer` command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

   ```
   $ sudo coreos-installer install \
   --ignition-url=https://host/worker.ign /dev/sda
   ```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.

IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

8.2.8.1.1. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

Routing and bonding options at RHCOS boot prompt

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.
IMPORTANT

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.

The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| To configure an IP address, either use DHCP (`ip=dhcp`) or set an individual static IP address (`ip=<host_ip>`). Then identify the DNS server IP address (`nameserver=<dns_ip>`) on each node. This example sets: | `ip=10.10.10.2::10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41` |
| ● The node’s IP address to **10.10.10.2** |  |
| ● The gateway address to **10.10.10.254** |  |
| ● The netmask to **255.255.255.0** |  |
| ● The hostname to **core0.example.com** |  |
| ● The DNS server address to **4.4.4.41** |  |
| Specify multiple network interfaces by specifying multiple `ip=` entries. | `ip=10.10.10.2::10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.254:255.255.255.0:core0.example.com:enp2s0:none` |
<p>| Optional: You can configure routes to additional networks by setting an <code>rd.route=</code> value. | To configure the default gateway: |
| If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway. | <code>ip=:10.10.10.254:::</code> |
| To configure the route for the additional network: | <code>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</code> |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more</td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</td>
</tr>
<tr>
<td>network interfaces and only one interface is being used.</td>
<td>ip=:::core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>You can combine DHCP and static IP configurations on systems with</td>
<td>ip=enp1s0:dhcp</td>
</tr>
<tr>
<td>multiple network interfaces.</td>
<td>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</td>
</tr>
<tr>
<td>Optional: You can configure VLANs on individual interfaces by using the</td>
<td>To configure a VLAN on a network interface and use a static IP address:</td>
</tr>
</tbody>
</table>
| vlan= parameter.                                                          | ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:enp2s0
| vlan=enp2s0.100:enp2s0                                                   | To configure a VLAN on a network interface and to use DHCP:             |
| You can provide multiple DNS servers by adding a nameserver= entry for     | nameserver=1.1.1.1
<p>| each server.                                                              | nameserver=8.8.8.8                                                      |
| Optional: Bonding multiple network interfaces to a single interface is     | To configure the bonded interface to use DHCP, set the bond’s IP address|
| supported using the bond= option.                                         | to dhcp.                                                                |
| In these two examples:                                                     | bond=bond0:em1,em2:mode=active-backup                                   |
| • The syntax for configuring a bonded interface is:                        | ip=bond0:dhcp                                                           |
| bond=name[:network_interfaces] [:options]                                 | To configure the bonded interface to use a static IP address, enter     |
| • name is the bonding device name (bond0),                              | bond=bond0:em1,em2:mode=active-backup                                   |
| network_interfaces represents a comma-separated list of physical (ethernet)| ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none |
| interfaces (em1,em2), and options is a comma-separated list of bonding    |                                                                         |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| Optional: You can configure VLANs on bonded interfaces by using the `vlan=` parameter. | To configure the bonded interface with a VLAN and to use DHCP: \[
|                                                                             |  \[ip=bond0.100:dhcp\] \n|                                                                             |  \[bond=bond0:em1,em2:mode=active-backup\] \n|                                                                             |  \[vlan=bond0.100:bond0\] \n|                                                                             | To configure the bonded interface with a VLAN and to use a static IP address: \[
|                                                                             |  \[ip=10.10.10.2::10.10.254:255.255.255.0:core0.example.com:bond0.100:none\] \n|                                                                             |  \[bond=bond0:em1,em2:mode=active-backup\] \n|                                                                             |  \[vlan=bond0.100:bond0\] \n|                                                                             |                                                                                           |
| Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example: | To configure a network team: \[
|                                                                             |  \[team=team0:em1,em2\] \n|                                                                             |  \[ip=team0:dhcp\] \n|                                                                             |                                                                                           |
|                                                                             |                                                                                           |
| **NOTE**                                                                    |                                                                                           |
|                                                                             | Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this Red Hat Knowledgebase Article. |

### 8.2.8.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Configure suitable PXE or iPXE infrastructure.
- Have access to an HTTP server that you can access from your computer.
Procedure

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**
   
   You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS **kernel**, **initramfs** and **rootfs** files from the RHCOS image mirror page.

   **IMPORTANT**
   
   The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

   The file names contain the OpenShift Container Platform version number. They resemble the following examples:

   - **kernel**: rhcos-<version>-live-kernel-<architecture>
   - **initramfs**: rhcos-<version>-live-initramfs.<architecture>.img
   - **rootfs**: rhcos-<version>-live-rootfs.<architecture>.img

3. Upload the additional files that are required for your booting method:

   - For traditional PXE, upload the **kernel** and **initramfs** files to your TFTP server and the **rootfs** file to your HTTP server.
   - For iPXE, upload the **kernel**, **initramfs**, and **rootfs** files to your HTTP server.

   **IMPORTANT**
   
   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images.

   Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

   - For PXE:
     
     DEFAULT pxedefault
     TIMEOUT 20
PROMPT 0
LABEL pxeboot

KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture>
coresos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign

1 Specify the location of the live kernel file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

2 If you use multiple NICs, specify a single interface in the ip option. For example, to use DHCP on a NIC that is named eno1, set ip=eno1:dhcp.

3 Specify locations of the RHCOS files that you uploaded to your HTTP server. The initrd parameter value is the location of the initramfs file, the coreos.live.rootsfs_url parameter value is the location of the rootsfs file, and the coreos.inst.ignition_url parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the APPEND line to configure networking or other boot options.

NOTE
This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more console= arguments to the APPEND line. For example, add console=tty0 console=ttyS0 to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?

For iPXE:

kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
<architecture>.img coreos.inst.install_dev=/dev/sda
coresos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
<architecture>.img

1 Specify locations of the RHCOS files that you uploaded to your HTTP server. The kernel parameter value is the location of the kernel file, the initrd=main argument is needed for booting on UEFI systems, the coreos.live.rootsfs_url parameter value is the location of the rootsfs file, and the coreos.inst.ignition_url parameter value is the location of the bootstrap Ignition config file.

2 If you use multiple NICs, specify a single interface in the ip option. For example, to use DHCP on a NIC that is named eno1, set ip=eno1:dhcp.

3 Specify the location of the initramfs file that you uploaded to your HTTP server.
NOTE

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more `console=` arguments to the `kernel` line. For example, add `console=tty0` `console=ttyS0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?.

6. If you use PXE UEFI, perform the following actions:
   a. Provide the `shimx64.efi` and `grubx64.efi` EFI binaries and the `grub.cfg` file that are required for booting the system.
      - Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the `images/efiboot.img` file to your host:

```
$ mkdir -p /mnt/iso
$ mkdir -p /mnt/efiboot
$ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
$ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
```

   - From the `efiboot.img` mount point, copy the `EFI/redhat/shimx64.efi` and `EFI/redhat/grubx64.efi` files to your TFTP server:

```
$ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
$ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
$ umount /mnt/efiboot
$ umount /mnt/iso
```

   - Copy the `EFI/redhat/grub.cfg` file that is included in the RHCOS ISO to your TFTP server.

   b. Edit the `grub.cfg` file to include arguments similar to the following:

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
  linuxefi rhcos=<version>-live-kernel=<architecture> coreos.inst.install_dev=/dev/sda
  <architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
  initrdelf rhcos=<version>-live-initramfs.<architecture>.img
}
```

where:
rhcos-<version>-live-kernel-<architecture>
Specifies the kernel file that you uploaded to your TFTP server.

Specifies the location of the live rootfs image that you uploaded to your HTTP server.

http://<HTTP_server>/bootstrap.ign
Specifies the location of the bootstrap Ignition config file that you uploaded to your HTTP server.

rhcos-<version>-live-initramfs.-architecture.-img
Specifies the location of the initramfs file that you uploaded to your TFTP server.

**NOTE**
For more information on how to configure a PXE server for UEFI boot, see the Red Hat Knowledgebase article: How to configure/setup a PXE server for UEFI boot for Red Hat Enterprise Linux?

7. Continue to create the machines for your cluster.

**IMPORTANT**
You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

## 8.2.9. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.

**Procedure**

1. Monitor the bootstrap process:

   ```
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \
   --log-level=info
   ```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`. 
Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

IMPORTANT

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

8.2.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami

   Example output

   system:admin

8.2.11. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.
Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   $ oc get nodes

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   $ oc get csr

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:
NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name>
  ```

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs --no-run-if-empty oc adm certificate approve
  ```

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
</tbody>
</table>
5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name>  
  ```

  * `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template='{range .items}{if not .status}{.metadata.name}{"\n"}{end}{end}' | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

  ```
  $ oc get nodes
  ```

**Example output**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

### 8.2.12. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

  ```
  $ watch -n5 oc get clusteroperators
  ```
2. Configure the Operators that are not available.

8.2.12.1. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

```bash
$ oc patch OperatorHub cluster --type json \
    -p '[["op": "add", "path": "/spec/disableAllDefaultSources", "value": true]]'
``
8.2.12.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

8.2.12.2.1. Changing the image registry’s management state

To start the image registry, you must change the Image Registry Operator configuration’s `managementState` from **Removed** to **Managed**.

Procedure

- Change `managementState` Image Registry Operator configuration from **Removed** to **Managed**. For example:

  
  ```shell
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"managementState":"Managed"}}'
  ```

8.2.12.2.2. Configuring registry storage for IBM Power Systems

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on IBM Power Systems.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have 100Gi capacity.

**Procedure**
1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:
   
   ```bash
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:
   
   ```bash
   $ oc edit configs.imagerepository.operator.openshift.io
   ```

   **Example output**

   ```json
   storage:
   pvc:
   claim:
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:
   
   ```bash
   $ oc get clusteroperator image-registry
   ```

5. Ensure that your registry is set to managed to enable building and pushing of images.
   
   - Run:
     
     ```bash
     $ oc edit configs.imagerepository/operator.openshift.io
     ```

     Then, change the line

     ```yaml
     managementState: Removed
     ```

     to

     ```yaml
     managementState: Managed
     ```

**8.2.12.2.3. Configuring storage for the image registry in non-production clusters**

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.
Procedure

- To set the image registry storage to an empty directory:

  ```
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"storage":{"emptyDir":{}}}}'
  ```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

8.2.13. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

Prerequisites

- Your control plane has initialized.

- You have completed the initial Operator configuration.

Procedure

1. Confirm that all the cluster components are online with the following command:

   ```
   $ watch -n5 oc get clusteroperators
   ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$. openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```bash
   $ oc get pods --all-namespaces
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 9m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 3m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 1m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 2m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 0 5m</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```bash
   $ oc logs <pod_name> -n <namespace>
   ```

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

---

### 8.2.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](https://docs.openshift.com/container-platform/4.6/cluster管理员/). After you confirm that your [OpenShift Cluster Manager](https://docs.openshift.com/container-platform/4.6/cluster管理员/) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, [use subscription watch](https://docs.openshift.com/container-platform/4.6/cluster管理员/subscription-watch.html) to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](https://docs.openshift.com/container-platform/4.6/cluster管理员/remote-health-monitoring.html) for more information about the Telemetry service

### 8.2.15. Next steps

- Customize your cluster.

- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
CHAPTER 9. INSTALLING ON OPENSTACK

9.1. INSTALLING A CLUSTER ON OPENSTACK WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a customized cluster on Red Hat OpenStack Platform (RHOSP). To customize the installation, modify parameters in the `install-config.yaml` before you install the cluster.

9.1.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
  - Verify that OpenShift Container Platform 4.6 is compatible with your RHOSP version in the Available platforms section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
- Verify that your network configuration does not rely on a provider network. Provider networks are not supported.
- Have a storage service installed in RHOSP, like block storage (Cinder) or object storage (Swift). Object storage is the recommended storage technology for OpenShift Container Platform registry cluster deployment. For more information, see Optimizing storage.
- Have metadata service enabled in RHOSP

9.1.2. Resource guidelines for installing OpenShift Container Platform on RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating IP addresses</td>
<td>3</td>
</tr>
<tr>
<td>Ports</td>
<td>15</td>
</tr>
<tr>
<td>Routers</td>
<td>1</td>
</tr>
<tr>
<td>Subnets</td>
<td>1</td>
</tr>
<tr>
<td>RAM</td>
<td>112 GB</td>
</tr>
<tr>
<td>vCPUs</td>
<td>28</td>
</tr>
<tr>
<td>Volume storage</td>
<td>275 GB</td>
</tr>
<tr>
<td>Instances</td>
<td>7</td>
</tr>
</tbody>
</table>
A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

### IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the `swiftoperator` role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

### NOTE

By default, your security group and security group rule quotas might be low. If you encounter problems, run `openstack quota set --secgroups 3 --secgroup-rules 60` as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

#### 9.1.2.1. Control plane and compute machines

By default, the OpenShift Container Platform installation process stands up three control plane and three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

### TIP

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

#### 9.1.2.2. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
• A port from the RHOSP quota
• A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

9.1.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

• Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

• Access Quay.io to obtain the packages that are required to install your cluster.

• Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

9.1.4. Enabling Swift on RHOSP

Swift is operated by a user account with the swiftoperator role. Add the role to an account before you run the installation program.

**IMPORTANT**

If the Red Hat OpenStack Platform (RHOSP) object storage service, commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.

**Prerequisites**

• You have a RHOSP administrator account on the target environment.

• The Swift service is installed.

• On Ceph RGW, the **account in url** option is enabled.

**Procedure**

To enable Swift on RHOSP:
1. As an administrator in the RHOSP CLI, add the `swiftoperator` role to the account that will access Swift:

   ```
   $ openstack role add --user <user> --project <project> swiftoperator
   ```

   Your RHOSP deployment can now use Swift for the image registry.

### 9.1.5. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

**Prerequisites**

- Configure OpenStack’s networking service to have DHCP agents forward instances’ DNS queries

**Procedure**

1. Using the RHOSP CLI, verify the name and ID of the ‘External’ network:

   ```
   $ openstack network list --long -c ID -c Name -c "Router Type"
   ```

   **Example output**

   ```
   +--------------------------------------+----------------+-------------+ |
   | ID                                   | Name           | Router Type |
   +--------------------------------------+----------------+-------------+ |
   | 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
   +--------------------------------------+----------------+-------------+ |
   ```

   A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network.

**IMPORTANT**

If the external network’s CIDR range overlaps one of the default network ranges, you must change the matching network ranges in the `install-config.yaml` file before you start the installation process.

The default network ranges are:

<table>
<thead>
<tr>
<th>Network</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineNetwork</td>
<td>10.0.0.0/16</td>
</tr>
<tr>
<td>serviceNetwork</td>
<td>172.30.0.0/16</td>
</tr>
<tr>
<td>clusterNetwork</td>
<td>10.128.0.0/14</td>
</tr>
</tbody>
</table>
WARNING

If the installation program finds multiple networks with the same name, it sets one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

NOTE

If the Neutron trunk service plug-in is enabled, a trunk port is created by default. For more information, see Neutron trunk port.

9.1.6. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called clouds.yaml. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the clouds.yaml file:

   - If your RHOSP distribution includes the Horizon web UI, generate a clouds.yaml file in it.

     **IMPORTANT**

     Remember to add a password to the auth field. You can also keep secrets in a separate file from clouds.yaml.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about clouds.yaml, see Config files in the RHOSP documentation.


```yaml
clouds:
  shiftstack:
    auth:
      project_name: shiftstack
      username: shiftstack_user
      password: XXX
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: 'devuser'
      password: XXX
      project_name: 'devonly'
```
2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
   a. Copy the certificate authority file to your machine.
   b. Add the machine to the certificate authority trust bundle:
      
      ```
      $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
      ```
   c. Update the trust bundle:
      
      ```
      $ sudo update-ca-trust extract
      ```
   d. Add the `cacerts` key to the `clouds.yaml` file. The value must be an absolute, non-root-accessible path to the CA certificate:
      
      ```
      clouds:
        shiftstack:
          ...
        cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
      ```

      **TIP**

      After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the `ca-cert.pem` key in the `cloud-provider-config` keymap. On a command line, run:

      ```
      $ oc edit configmap -n openshift-config cloud-provider-config
      ```

3. Place the `clouds.yaml` file in one of the following locations:
   a. The value of the `OS_CLIENT_CONFIG_FILE` environment variable
   b. The current directory
   c. A Unix-specific user configuration directory, for example `~/.config/openstack/clouds.yaml`
   d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`

      The installation program searches for `clouds.yaml` in that order.

### 9.1.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 9.1.8. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   **For `<installation_directory>`**, specify the directory name to store the files that the installation program creates.
b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `openstack` as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane and compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the *Installation configuration parameters* section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 9.1.8.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**
- You have an existing `install-config.yaml` file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----

   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   2 A proxy URL to use for creating HTTPS connections outside the cluster.

   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RH COS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RH COS trust bundle.
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 9.1.9. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 9.1.9.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 9.2. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiVersion</code></td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as <code>dev</code>. The string must be 14 characters or fewer long.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auths&quot;:{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;cloud.openshift.com&quot;:{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth&quot;:&quot;b3Blb=&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>},</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;quay.io&quot;:{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;auth&quot;:&quot;b3Blb=&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
### 9.1.9.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 9.3. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>You cannot modify parameters specified by the networking object after installation.</td>
<td></td>
</tr>
<tr>
<td>networking.networkType</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.128.0.0/14 with a host prefix of /23.</td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An IPv4 network.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default value is 23.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

| networking.serviceNetwork | The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network. | An array with an IP address block in CIDR format. For example: `networking:  
  serviceNetwork:  
  - 172.30.0.0/16` |
| networking.machineNetwork | The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. | An array of objects. For example: `networking:  
  machineNetwork:  
  - cidr: 10.0.0.0/16` |
| networking.machineNetwork.cidr | Required if you use `networking.machineNetwork`. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24. | An IP network block in CIDR notation. For example, 10.0.0.0/16. |

#### NOTE
Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

### 9.1.9.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

#### Table 9.4. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><code>compute.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or{}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>controlPlane.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><strong>controlPlane.hypertreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td><strong>controlPlane.name</strong></td>
<td>Required if you use <strong>controlPlane</strong>. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td><strong>controlPlane.platform</strong></td>
<td>Required if you use <strong>controlPlane</strong>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <strong>compute.platform</strong> parameter value.</td>
<td><strong>aws</strong>, <strong>azure</strong>, <strong>gcp</strong>, <strong>openstack</strong>, <strong>ovirt</strong>, <strong>vsphere</strong>, or {}</td>
</tr>
<tr>
<td><strong>controlPlane.replicas</strong></td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.</td>
<td></td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</td>
</tr>
</tbody>
</table>

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

### 9.1.9.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

Table 9.5. Additional RHOSP parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.rootVolume.size</code></td>
<td>For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td><code>compute.platform.openstack.rootVolume.type</code></td>
<td>For compute machines, the root volume’s type.</td>
<td>String, for example <code>performance</code>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.rootVolume.size</code></td>
<td>For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.rootVolume.type</code></td>
<td>For control plane machines, the root volume’s type.</td>
<td>String, for example <code>performance</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.cloud</code></td>
<td>The name of the RHOSP cloud to use from the list of clouds in the <code>clouds.yaml</code> file.</td>
<td>String, for example <code>MyCloud</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.externalNetwork</code></td>
<td>The RHOSP external network name to be used for installation.</td>
<td>String, for example <code>external</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.computeFlavor</code></td>
<td>The RHOSP flavor to use for control plane and compute machines.</td>
<td>String, for example <code>m1.xlarge</code>.</td>
</tr>
</tbody>
</table>

### 9.1.9.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 9.6. Optional RHOSP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.additionalNetworkIds</code></td>
<td>Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>compute.platform.openstack.additionalSecurityGroupIDs</td>
<td>Additional security groups that are associated with compute machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>compute.platform.openstack.zones</td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.additionalNetworkIDs</td>
<td>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.additionalSecurityGroupIDs</td>
<td>Additional security groups that are associated with control plane machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.zones</td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td>platform.openstack.clusterOSImage</td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with an SHA-256 checksum. For example, <a href="http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d">http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</a>. The value can also be the name of an existing Glance image, for example my-rhcos.</td>
</tr>
<tr>
<td>platform.openstack.defaultMachinePlatform</td>
<td>The default machine pool platform configuration.</td>
<td><code>{  &quot;type&quot;: &quot;ml.large&quot;,  &quot;rootVolume&quot;: {  &quot;size&quot;: 30,  &quot;type&quot;: &quot;performance&quot; } }</code></td>
</tr>
<tr>
<td>platform.openstack.ingressFloatingIP</td>
<td>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.openstack.lbFloatingIP</td>
<td>An existing floating IP address to associate with the API load balancer. To use this property, you must also define the <strong>platform.openstack.externalNetwork</strong> property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>platform.openstack.externalDNS</td>
<td>IP addresses for external DNS servers that cluster instances use for DNS resolution.</td>
<td>A list of IP addresses as strings. For example, [&quot;8.8.8.8&quot;, &quot;192.168.1.12&quot;].</td>
</tr>
<tr>
<td>platform.openstack.machinesSubnet</td>
<td>The UUID of a RHOSP subnet that the cluster’s nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.</td>
<td>A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
</tbody>
</table>

#### 9.1.9.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet’s GUID is passed as the value of **platform.openstack.machinesSubnet** in the `install-config.yaml` file.

This subnet is used as the cluster’s primary subnet; nodes and ports are created on it.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that:

- The target network and subnet are available.
- DHCP is enabled on the target subnet.
- You can provide installer credentials that have permission to create ports on the target network.
- If your network configuration requires a router, it is created in RHOSP. Some configurations rely on routers for floating IP address translation.
Your network configuration does not rely on a provider network. Provider networks are not supported.

**NOTE**

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network’s CIDR block. To override these default values, set values for `platform.openstack.apiVIP` and `platform.openstack.ingressVIP` that are outside of the DHCP allocation pool.

### 9.1.9.7. Sample customized install-config.yaml file for RHOSP

This sample `install-config.yaml` demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

**IMPORTANT**

This sample file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program.

```yaml
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
  - name: worker
    platform:
      openstack:
        type: ml.large
        replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  serviceNetwork:
    - 172.30.0.0/16
networkType: OpenShiftSDN
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```
9.1.10. Setting compute machine affinity

Optionally, you can set the affinity policy for compute machines during installation. The installer does not select an affinity policy for compute machines by default.

You can also create machine sets that use particular RHOSP server groups after installation.

NOTE
Control plane machines are created with a soft-anti-affinity policy.

TIP
You can learn more about RHOSP instance scheduling and placement in the RHOSP documentation.

Prerequisites

- Create the install-config.yaml file and complete any modifications to it.

Procedure

1. Using the RHOSP command-line interface, create a server group for your compute machines. For example:

```bash
$ openstack \
--os-compute-api-version=2.15 \
server group create \
--policy anti-affinity \
my-openshift-worker-group
```

For more information, see the server group create command documentation.

2. Change to the directory that contains the installation program and create the manifests:

```bash
$ ./openshift-install create manifests --dir=<installation_directory>
```

where:

- **installation_directory**: Specifies the name of the directory that contains the install-config.yaml file for your cluster.

3. Open manifests/99_openshift-cluster-api_worker-machineset-0.yaml, the MachineSet definition file.

4. Add the property serverGroupId to the definition beneath the spec.template.spec.providerSpec.value property. For example:

```yaml
apiVersion: machine.openshift.io/v1beta1
dkind: MachineSet
dmetadata:
labels:
  machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
  machine.openshift.io/cluster-api-machine-role: <node_role>
```
Add the UUID of your server group here.

5. Optional: Back up the manifests/99_openshift-cluster-api_worker-machineset-0.yaml file. The installation program deletes the manifests/ directory when creating the cluster.

When you install the cluster, the installer uses the MachineSet definition that you modified to create compute machines within your RHOSP server group.
9.1.11. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
   ``

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>
   ```
Example output

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.1.12. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

9.1.12.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

   $ openstack floating ip create --description "API <cluster_name>.<base_domain>" <external_network>

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

   $ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>" <external_network>

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

   api.<cluster_name>.<base_domain>. IN A <API_FIP>
   *.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>

   NOTE

   If you do not control the DNS server, you can add the record to your /etc/hosts file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

4. Add the FIPs to the install-config.yaml file as the values of the following parameters:

   - platform.openstack.ingressFloatingIP
- platform.openstack.lbFloatingIP

If you use these values, you must also enter an external network as the value of the platform.openstack.externalNetwork parameter in the install-config.yaml file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

9.1.12.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the install-config.yaml file, do not define the following parameters:

- platform.openstack.ingressFloatingIP
- platform.openstack.lbFloatingIP

If you cannot provide an external network, you can also leave platform.openstack.externalNetwork blank. If you do not provide a value for platform.openstack.externalNetwork, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.

NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>

If you do not control the DNS server, you can add the record to your /etc/hosts file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

9.1.13. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites
Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \   --log-level=info
   
   1 For `<installation_directory>`, specify the location of your customized ./install-config.yaml file.

   2 To view different installation details, specify warn, debug, or error instead of info.

   NOTE
   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the kubeadmin user, display in your terminal.

   Example output

   ...
   INFO Install complete!
   INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
   INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
   INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
   INFO Time elapsed: 36m22s

   NOTE
   The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

9.1.14. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

Procedure

1. In the cluster environment, export the administrator’s kubeconfig file:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

   $ oc get nodes

3. View your cluster’s version:

   $ oc get clusterversion

4. View your Operators’ status:

   $ oc get clusteroperator

5. View all running pods in the cluster:

   $ oc get pods -A
9.1.15. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster *kubeconfig* file. The *kubeconfig* file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the *oc* CLI.

**Procedure**

1. Export the *kubeadmin* credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run *oc* commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   system:admin

**Additional resources**

- See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.1.16. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use *subscription watch* to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service

9.1.17. Next steps

- Customize your cluster.
If necessary, you can opt out of remote health reporting.

If you need to enable external access to node ports, configure ingress cluster traffic by using a node port.

If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses.

9.2. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR

In OpenShift Container Platform version 4.6, you can install a customized cluster on Red Hat OpenStack Platform (RHOSP) that uses Kuryr SDN. To customize the installation, modify parameters in the `install-config.yaml` before you install the cluster.

9.2.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
  - Verify that OpenShift Container Platform 4.6 is compatible with your RHOSP version in the Available platforms section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.

- Verify that your network configuration does not rely on a provider network. Provider networks are not supported.

- Have a storage service installed in RHOSP, like block storage (Cinder) or object storage (Swift). Object storage is the recommended storage technology for OpenShift Container Platform registry cluster deployment. For more information, see Optimizing storage.

9.2.2. About Kuryr SDN

Kuryr is a container network interface (CNI) plug-in solution that uses the Neutron and Octavia Red Hat OpenStack Platform (RHOSP) services to provide networking for pods and Services.

Kuryr and OpenShift Container Platform integration is primarily designed for OpenShift Container Platform clusters running on RHOSP VMs. Kuryr improves the network performance by plugging OpenShift Container Platform pods into RHOSP SDN. In addition, it provides interconnectivity between pods and RHOSP virtual instances.

Kuryr components are installed as pods in OpenShift Container Platform using the `openshift-kuryr` namespace:

- **kuryr-controller** - a single service instance installed on a master node. This is modeled in OpenShift Container Platform as a Deployment object.

- **kuryr-cni** - a container installing and configuring Kuryr as a CNI driver on each OpenShift Container Platform node. This is modeled in OpenShift Container Platform as a DaemonSet object.

The Kuryr controller watches the OpenShift Container Platform API server for pod, service, and namespace create, update, and delete events. It maps the OpenShift Container Platform API calls to corresponding objects in Neutron and Octavia. This means that every network solution that implements the Neutron trunk port functionality can be used to back OpenShift Container Platform via Kuryr. This includes open source solutions such as Open vSwitch (OVS) and Open Virtual Network (OVN) as well as Neutron-compatible commercial SDNs.
Kuryr is recommended for OpenShift Container Platform deployments on encapsulated RHOSP tenant networks to avoid double encapsulation, such as running an encapsulated OpenShift Container Platform SDN over an RHOSP network.

If you use provider networks or tenant VLANs, you do not need to use Kuryr to avoid double encapsulation. The performance benefit is negligible. Depending on your configuration, though, using Kuryr to avoid having two overlays might still be beneficial.

Kuryr is not recommended in deployments where all of the following criteria are true:

- The RHOSP version is less than 16.
- The deployment uses UDP services, or a large number of TCP services on few hypervisors.

or

- The `ovn-octavia` Octavia driver is disabled.
- The deployment uses a large number of TCP services on few hypervisors.

### 9.2.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr

When using Kuryr SDN, the pods, services, namespaces, and network policies are using resources from the RHOSP quota; this increases the minimum requirements. Kuryr also has some additional requirements on top of what a default install requires.

Use the following quota to satisfy a default cluster’s minimum requirements:

**Table 9.7. Recommended resources for a default OpenShift Container Platform cluster on RHOSP with Kuryr**

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating IP addresses</td>
<td>3 - plus the expected number of Services of LoadBalancer type</td>
</tr>
<tr>
<td>Ports</td>
<td>1500 - 1 needed per Pod</td>
</tr>
<tr>
<td>Routers</td>
<td>1</td>
</tr>
<tr>
<td>Subnets</td>
<td>250 - 1 needed per Namespace/Project</td>
</tr>
<tr>
<td>Networks</td>
<td>250 - 1 needed per Namespace/Project</td>
</tr>
<tr>
<td>RAM</td>
<td>112 GB</td>
</tr>
<tr>
<td>vCPUs</td>
<td>28</td>
</tr>
<tr>
<td>Volume storage</td>
<td>275 GB</td>
</tr>
<tr>
<td>Instances</td>
<td>7</td>
</tr>
</tbody>
</table>
A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

## IMPORTANT

If RHOSP object storage (Swift) is available and operated by a user account with the `swiftoperator` role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

## IMPORTANT

If you are using Red Hat OpenStack Platform (RHOSP) version 16 with the Amphora driver rather than the OVN Octavia driver, security groups are associated with service accounts instead of user projects.

Take the following notes into consideration when setting resources:

- The number of ports that are required is larger than the number of pods. Kuryr uses ports pools to have pre-created ports ready to be used by pods and speed up the pods' booting time.

- Each network policy is mapped into an RHOSP security group, and depending on the `NetworkPolicy` spec, one or more rules are added to the security group.

- Each service is mapped to an RHOSP load balancer. Consider this requirement when estimating the number of security groups required for the quota.
  
  If you are using RHOSP version 15 or earlier, or the `ovn-octavia` driver, each load balancer has a security group with the user project.

- The quota does not account for load balancer resources (such as VM resources), but you must consider these resources when you decide the RHOSP deployment’s size. The default installation will have more than 50 load balancers; the clusters must be able to accommodate them.
  
  If you are using RHOSP version 16 with the OVN Octavia driver enabled, only one load balancer VM is generated; services are load balanced through OVN flows.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.
To enable Kuryr SDN, your environment must meet the following requirements:

- Run RHOSP 13+.
- Have Overcloud with Octavia.
- Use Neutron Trunk ports extension.
- Use openvswitch firewall driver if ML2/OVS Neutron driver is used instead of ovs-hybrid.

### 9.2.3.1. Increasing quota

When using Kuryr SDN, you must increase quotas to satisfy the Red Hat OpenStack Platform (RHOSP) resources used by pods, services, namespaces, and network policies.

**Procedure**

- Increase the quotas for a project by running the following command:

```
$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets 250 --networks 250 <project>
```

### 9.2.3.2. Configuring Neutron

Kuryr CNI leverages the Neutron Trunks extension to plug containers into the Red Hat OpenStack Platform (RHOSP) SDN, so you must use the trunks extension for Kuryr to properly work.

In addition, if you leverage the default ML2/OVS Neutron driver, the firewall must be set to openvswitch instead of ovs_hybrid so that security groups are enforced on trunk subports and Kuryr can properly handle network policies.

### 9.2.3.3. Configuring Octavia

Kuryr SDN uses Red Hat OpenStack Platform (RHOSP)’s Octavia LBaaS to implement OpenShift Container Platform services. Thus, you must install and configure Octavia components in RHOSP to use Kuryr SDN.

To enable Octavia, you must include the Octavia service during the installation of the RHOSP Overcloud, or upgrade the Octavia service if the Overcloud already exists. The following steps for enabling Octavia apply to both a clean install of the Overcloud or an Overcloud update.

**NOTE**

The following steps only capture the key pieces required during the deployment of RHOSP when dealing with Octavia. It is also important to note that registry methods vary.

This example uses the local registry method.

**Procedure**

1. If you are using the local registry, create a template to upload the images to the registry. For example:

```
(undercloud) $ openstack overcloud container image prepare
```
2. Verify that the `local_registry_images.yaml` file contains the Octavia images. For example:

```
... 
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
  push_destination: <local-ip-from-undercloud.conf>:8787
- imagename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
  push_destination: <local-ip-from-undercloud.conf>:8787
```

**NOTE**

The Octavia container versions vary depending upon the specific RHOSP release installed.

3. Pull the container images from `registry.redhat.io` to the Undercloud node:

```
(undercloud) $ sudo openstack overcloud container image upload \
  --config-file /home/stack/local_registry_images.yaml \
  --verbose
```

This may take some time depending on the speed of your network and Undercloud disk.

4. Since an Octavia load balancer is used to access the OpenShift Container Platform API, you must increase their listeners' default timeouts for the connections. The default timeout is 50 seconds. Increase the timeout to 20 minutes by passing the following file to the Overcloud deploy command:

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```

**NOTE**

This is not needed for RHOSP 13.0.13+.

5. Install or update your Overcloud environment with Octavia:

```
$ openstack overcloud deploy --templates \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
  -e octavia_timeouts.yaml
```
NOTE

This command only includes the files associated with Octavia; it varies based on your specific installation of RHOSP. See the RHOSP documentation for further information. For more information on customizing your Octavia installation, see installation of Octavia using Director.

NOTE

When leveraging Kuryr SDN, the Overcloud installation requires the Neutron trunk extension. This is available by default on director deployments. Use the openvswitch firewall instead of the default ovs-hybrid when the Neutron backend is ML2/OVS. There is no need for modifications if the backend is ML2/OVN.

6. In RHOSP versions earlier than 13.0.13, add the project ID to the octavia.conf configuration file after you create the project.

   - To enforce network policies across services, like when traffic goes through the Octavia load balancer, you must ensure Octavia creates the Amphora VM security groups on the user project. This change ensures that required load balancer security groups belong to that project, and that they can be updated to enforce services isolation.

   NOTE

   This task is unnecessary in RHOSP version 13.0.13 or later.

Octavia implements a new ACL API that restricts access to the load balancers VIP.

a. Get the project ID

   $ openstack project show <project>

   Example output

   +-------------+----------------------------------+
   | Field       | Value                            |
   +-------------+----------------------------------+
   | description |                                  |
   | domain_id   | default                          |
   | enabled     | True                             |
   | id          | PROJECT_ID                       |
   | is_domain   | False                            |
   | name        | *<project>*                      |
   | parent_id   | default                          |
   | tags        | []                               |
   +-------------+----------------------------------+

b. Add the project ID to octavia.conf for the controllers.

   i. Source the stackrc file:
$ source stackrc  # Undercloud credentials

ii. List the Overcloud controllers:

$ openstack server list

Example output

+--------------------------------------+--------------+--------+-----------------------+-------
| ID                                   | Name         | Status | Networks
| Image          | Flavor       |
|----------------+--------------+--------+-----------------------+-------|
+--------------------------------------+--------------+--------+-----------------------+-------|
| 6bef8e73-2ba5-4860-a0b1-3937f8ca7e01 | controller-0 | ACTIVE | overcloud-full        |
| dda3173a-ab26-47f8-a2dc-8473b4a67ab9 | compute-0    | ACTIVE | overcloud-full        |
+--------------------------------------+--------------+--------+-----------------------+-------|

iii. SSH into the controller(s).

$ ssh heat-admin@192.168.24.8

c. Restart the Octavia worker so the new configuration loads.

controller-0$ sudo docker restart octavia_worker

iv. Edit the `octavia.conf` file to add the project into the list of projects where Amphora security groups are on the user’s account.

```
# List of project IDs that are allowed to have Load balancer security groups
# belonging to them.
amp_secgroup_allowed_projects = PROJECT_ID
```

NOTE

Depending on your RHOSP environment, Octavia might not support UDP listeners. If you use Kuryr SDN on RHOSP version 13.0.13 or earlier, UDP services are not supported. RHOSP version 16 or later support UDP.

9.2.3.3.1. The Octavia OVN Driver

Octavia supports multiple provider drivers through the Octavia API.

To see all available Octavia provider drivers, on a command line, enter:
Beginning with RHOSP version 16, the Octavia OVN provider driver (ovn) is supported on OpenShift Container Platform on RHOSP deployments.

*ovn* is an integration driver for the load balancing that Octavia and OVN provide. It supports basic load balancing capabilities, and is based on OpenFlow rules. The driver is automatically enabled in Octavia by Director on deployments that use OVN Neutron ML2.

The Amphora provider driver is the default driver. If *ovn* is enabled, however, Kuryr uses it.

If Kuryr uses *ovn* instead of Amphora, it offers the following benefits:

- Decreased resource requirements. Kuryr does not require a load balancer VM for each service.
- Reduced network latency.
- Increased service creation speed by using OpenFlow rules instead of a VM for each service.
- Distributed load balancing actions across all nodes instead of centralized on Amphora VMs.

You can configure your cluster to use the Octavia OVN driver after your RHOSP cloud is upgraded from version 13 to version 16.

### 9.2.3.4. Known limitations of installing with Kuryr

Using OpenShift Container Platform with Kuryr SDN has several known limitations.

#### RHOSP general limitations

OpenShift Container Platform with Kuryr SDN does not support *Service* objects with type *NodePort*.

If the machines subnet is not connected to a router, or if the subnet is connected, but the router has no external gateway set, Kuryr cannot create floating IPs for *Service* objects with type *LoadBalancer*.

- Configuring the *sessionAffinity=ClientIP* property on *Service* objects does not have an effect. Kuryr does not support this setting.

#### RHOSP version limitations

Using OpenShift Container Platform with Kuryr SDN has several limitations that depend on the RHOSP version.

- RHOSP versions before 16 use the default Octavia load balancer driver (Amphora). This driver requires that one Amphora load balancer VM is deployed per OpenShift Container Platform service. Creating too many services can cause you to run out of resources.
Deployments of later versions of RHOSP that have the OVN Octavia driver disabled also use the Amphora driver. They are subject to the same resource concerns as earlier versions of RHOSP.

- Octavia RHOSP versions before 13.0.13 do not support UDP listeners. Therefore, OpenShift Container Platform UDP services are not supported.

- Octavia RHOSP versions before 13.0.13 cannot listen to multiple protocols on the same port. Services that expose the same port to different protocols, like TCP and UDP, are not supported.

- Kuryr SDN does not support automatic unidling by a service.

**RHOSP environment limitations**

There are limitations when using Kuryr SDN that depend on your deployment environment.

Because of Octavia’s lack of support for the UDP protocol and multiple listeners, if the RHOSP version is earlier than 13.0.13, Kuryr forces pods to use TCP for DNS resolution.

In Go versions 1.12 and earlier, applications that are compiled with CGO support disabled use UDP only. In this case, the native Go resolver does not recognize the `use-vc` option in `resolv.conf`, which controls whether TCP is forced for DNS resolution. As a result, UDP is still used for DNS resolution, which fails.

To ensure that TCP forcing is allowed, compile applications either with the environment variable `CGO_ENABLED` set to 1, i.e. `CGO_ENABLED=1`, or ensure that the variable is absent.

In Go versions 1.13 and later, TCP is used automatically if DNS resolution using UDP fails.

**NOTE**

musl-based containers, including Alpine-based containers, do not support the `use-vc` option.

**RHOSP upgrade limitations**

As a result of the RHOSP upgrade process, the Octavia API might be changed, and upgrades to the Amphora images that are used for load balancers might be required.

You can address API changes on an individual basis.

If the Amphora image is upgraded, the RHOSP operator can handle existing load balancer VMs in two ways:

- Upgrade each VM by triggering a load balancer failover.

- Leave responsibility for upgrading the VMs to users.

If the operator takes the first option, there might be short downtimes during failovers.

If the operator takes the second option, the existing load balancers will not support upgraded Octavia API features, like UDP listeners. In this case, users must recreate their Services to use these features.
If OpenShift Container Platform detects a new Octavia version that supports UDP load balancing, it recreates the DNS service automatically. The service recreation ensures that the service default supports UDP load balancing.

The recreation causes the DNS service approximately one minute of downtime.

### 9.2.3.5. Control plane and compute machines

By default, the OpenShift Container Platform installation process stands up three control plane and three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

**TIP**

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

### 9.2.3.6. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

### 9.2.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](#) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](#) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

9.2.5. Enabling Swift on RHOSP

Swift is operated by a user account with the `swiftoperator` role. Add the role to an account before you run the installation program.

IMPORTANT

If the Red Hat OpenStack Platform (RHOSP) object storage service, commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.

Prerequisites

- You have a RHOSP administrator account on the target environment.
- The Swift service is installed.
- On Ceph RGW, the `account in url` option is enabled.

Procedure

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the `swiftoperator` role to the account that will access Swift:

   ```
   $ openstack role add --user <user> --project <project> swiftoperator
   ```

   Your RHOSP deployment can now use Swift for the image registry.

9.2.6. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack’s networking service to have DHCP agents forward instances’ DNS queries
Procedure

1. Using the RHOSP CLI, verify the name and ID of the 'External' network:

   ```
   $ openstack network list --long -c ID -c Name -c "Router Type"
   ```

   **Example output**

   ```
   +--------------------------------------+----------------+-------------+
   | ID                                   | Name           | Router Type |
   +--------------------------------------+----------------+-------------+
   | 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
   +--------------------------------------+----------------+-------------+
   ```

   A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network.

   **IMPORTANT**

   If the external network’s CIDR range overlaps one of the default network ranges, you must change the matching network ranges in the `install-config.yaml` file before you start the installation process.

   The default network ranges are:

<table>
<thead>
<tr>
<th>Network</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>machineNetwork</code></td>
<td>10.0.0.0/16</td>
</tr>
<tr>
<td><code>serviceNetwork</code></td>
<td>172.30.0.0/16</td>
</tr>
<tr>
<td><code>clusterNetwork</code></td>
<td>10.128.0.0/14</td>
</tr>
</tbody>
</table>

   **WARNING**

   If the installation program finds multiple networks with the same name, it sets one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

   **NOTE**

   If the Neutron trunk service plug-in is enabled, a trunk port is created by default. For more information, see Neutron trunk port.

9.2.7. Defining parameters for the installation program
The OpenShift Container Platform installation program relies on a file that is called `clouds.yaml`. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, login information, and authorization service URLs.

**Procedure**

1. Create the `clouds.yaml` file:
   - If your RHOSP distribution includes the Horizon web UI, generate a `clouds.yaml` file in it.
     - IMPORTANT
       Remember to add a password to the `auth` field. You can also keep secrets in a separate file from `clouds.yaml`.
     - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about `clouds.yaml`, see Config files in the RHOSP documentation.

   ```yaml
   clouds:
     shiftstack:
       auth:
         project_name: shiftstack
         username: shiftstack_user
         password: XXX
         user_domain_name: Default
         project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: 'devuser'
           password: XXX
           project_name: 'devonly'
   
   clouds:
     shiftstack:
   
   $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
   $ sudo update-ca-trust extract
   
   d. Add the `cacerts` key to the `clouds.yaml` file. The value must be an absolute, non-root-accessible path to the CA certificate:

   ```yaml
   clouds:
     shiftstack:
   ```
TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the `ca-cert.pem` key in the `cloud-provider-config` keymap. On a command line, run:

```bash
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the `clouds.yaml` file in one of the following locations:
   a. The value of the `OS_CLIENT_CONFIG_FILE` environment variable
   b. The current directory
   c. A Unix-specific user configuration directory, for example `~/.config/openstack/clouds.yaml`
   d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`

   The installation program searches for `clouds.yaml` in that order.

9.2.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.
Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.2.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create the `install-config.yaml` file.
   a. Change to the directory that contains the installation program and run the following command:

      $ ./openshift-install create install-config --dir <installation_directory>

      For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

      IMPORTANT

      Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.
NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select openstack as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane and compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the install-config.yaml file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the install-config.yaml file so that you can use it to install multiple clusters.

IMPORTANT

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.2.9.1. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the install-config.yaml file.

Prerequisites

- You have an existing install-config.yaml file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRusted_CA_CERT>
     -----END CERTIFICATE-----
   ...
   ```

   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   2 A proxy URL to use for creating HTTPS connections outside the cluster.

   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations.

   4 If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RH COS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RH COS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 9.2.10. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

#### 9.2.10.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 9.8. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}. {{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev. The string must be 14 characters or fewer long.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
| pullSecret | Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io. | {
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
} |
9.2.10.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 9.9. Network parameters

| Parameter               | Description                                                                 | Values                                                                 |
|-------------------------|-----------------------------------------------------------------------------|                                                                      |
| networking               | The configuration for the cluster network.                                  | Object                                                               |
|                         | NOTE                                                                        | You cannot modify parameters specified by the networking object after installation. |
| networking.network      | The cluster network provider Container Network Interface (CNI) plug-in to install. | Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN. |
| networking.clusterNetwork | The IP address blocks for pods.                                             | An array of objects. For example:                                    |
|                         | The default value is 10.128.0.0/14 with a host prefix of /23.               | networking:                                                          |
|                         | If you specify multiple IP address blocks, the blocks must not overlap.     | clusterNetwork:                                                      |
|                         |                                                                           | - cidr: 10.128.0.0/14                                               |
|                         |                                                                           | hostPrefix: 23                                                       |
| networking.clusterNetwork.cidr | Required if you use networking.clusterNetwork. An IP address block.          | An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32. |
|                         |                                                                           | An IPv4 network.                                                     |
| networking.clusterNetwork.hostPrefix | The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses. | A subnet prefix.                                                     |
|                         |                                                                           | The default value is 23.                                             |
### networking.serviceNetwork

The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>networking:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machineNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
</tbody>
</table>

### 9.2.10.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

#### Table 9.10. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
| controlPlane.hypertreading| Whether to enable or disable simultaneous multithreading, or `hypertreading`, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.  

**IMPORTANT**  
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. | Enabled or Disabled |
<p>| controlPlane.name        | Required if you use <code>controlPlane</code>. The name of the machine pool.            | master                                      |
| controlPlane.platform    | Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value. | <code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code> |
| controlPlane.replicas    | The number of control plane machines to provision.                           | The only supported value is 3, which is the default value. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.</td>
<td></td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the <strong>x86_64</strong> architecture.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: sshKey: <code>&lt;key1&gt;</code> <code>&lt;key2&gt;</code> <code>&lt;key3&gt;</code></td>
</tr>
</tbody>
</table>

### IMPORTANT

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

### NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

#### 9.2.10.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

**Table 9.11. Additional RHOSP parameters**
### Table 9.12. Optional RHOSP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.openstack.computePlatform.additionalNetworkIds</td>
<td>Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>compute.platform.openstack.computerPlatform.rootVolume.size</td>
<td>For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td>compute.platform.openstack.computerPlatform.rootVolume.type</td>
<td>For compute machines, the root volume’s type.</td>
<td>String, for example performance.</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.controlPlane.rootVolume.size</td>
<td>For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.controlPlane.rootVolume.type</td>
<td>For control plane machines, the root volume’s type.</td>
<td>String, for example performance.</td>
</tr>
<tr>
<td>platform.openstack.cloud</td>
<td>The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.</td>
<td>String, for example MyCloud.</td>
</tr>
<tr>
<td>platform.openstack.externalNetwork</td>
<td>The RHOSP external network name to be used for installation.</td>
<td>String, for example external.</td>
</tr>
<tr>
<td>platform.openstack.computeFlavor</td>
<td>The RHOSP flavor to use for control plane and compute machines.</td>
<td>String, for example m1.xlarge.</td>
</tr>
</tbody>
</table>

### 9.2.10.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.openstack.additionalsecurity</td>
<td>Additional security groups that are associated with compute machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>GroupIDs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>compute.platform.openstack.zones</td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set,</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td></td>
<td>the installer relies on the default settings for Nova that the RHOSP administrator configured.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>if you are using the Amphora provider driver, OpenShift Container Platform services that rely on</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Amphora VMs, are not created according to the value of this property.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.platform.openstack.additionalNet</td>
<td>Additional networks that are associated with control plane machines. Allowed address pairs are not</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>workIDs</td>
<td>created for additional networks.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.platform.openstack.additionalsec</td>
<td>Additional security groups that are associated with control plane machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>urityGroupIDs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.platform.openstack.zones</td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td>platform.openstack.clusterOSImage</td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with an SHA-256 checksum. For example, <a href="http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d">http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</a>. The value can also be the name of an existing Glance image, for example my-rhcos.</td>
</tr>
<tr>
<td>platform.openstack.defaultMachinePlatform</td>
<td>The default machine pool platform configuration.</td>
<td>{}</td>
</tr>
<tr>
<td>platform.openstack.ingressFloatingIP</td>
<td>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.openstack.lbFloatingIP</td>
<td>An existing floating IP address to associate with the API load balancer. To use this property, you must also define the platform.openstack.externalNetwork property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>platform.openstack.externalDNS</td>
<td>IP addresses for external DNS servers that cluster instances use for DNS resolution.</td>
<td>A list of IP addresses as strings. For example, [&quot;8.8.8.8&quot;, &quot;192.168.1.12&quot;].</td>
</tr>
<tr>
<td>platform.openstack.machinesSubnet</td>
<td>The UUID of a RHOSP subnet that the cluster’s nodes use. Nodes and virtual IP (VIP) ports are created on this subnet. The first item in networking.machineNetwork must match the value of machinesSubnet. If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</td>
<td>A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
</tbody>
</table>

### 9.2.10.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet’s GUID is passed as the value of platform.openstack.machinesSubnet in the install-config.yaml file.

This subnet is used as the cluster’s primary subnet; nodes and ports are created on it.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that:

- The target network and subnet are available.
- DHCP is enabled on the target subnet.
- You can provide installer credentials that have permission to create ports on the target network.
- If your network configuration requires a router, it is created in RHOSP. Some configurations rely on routers for floating IP address translation.
Your network configuration does not rely on a provider network. Provider networks are not supported.

NOTE
By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network’s CIDR block. To override these default values, set values for platform.openstack.apiVIP and platform.openstack.ingressVIP that are outside of the DHCP allocation pool.

9.2.10.7. Sample customized install-config.yaml file for RHOSP with Kuryr

To deploy with Kuryr SDN instead of the default OpenShift SDN, you must modify the install-config.yaml file to include Kuryr as the desired networking.networkType and proceed with the default OpenShift Container Platform SDN installation steps. This sample install-config.yaml demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

IMPORTANT
This sample file is provided for reference only. You must obtain your install-config.yaml file by using the installation program.

```yaml
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
      replicas: 3
  metadata:
    name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: Kuryr
  platform:
    openstack:
      cloud: mycloud
      externalNetwork: external
      computeFlavor: m1.xlarge
  lbFloatingIP: 128.0.0.1
  trunkSupport: true
```
The Amphora Octavia driver creates two ports per load balancer. As a result, the service subnet that the installer creates is twice the size of the CIDR that is specified as the value of the `serviceNetwork` property. The larger range is required to prevent IP address conflicts.

Both `trunkSupport` and `octaviaSupport` are automatically discovered by the installer, so there is no need to set them. But if your environment does not meet both requirements, Kuryr SDN will not properly work. Trunks are needed to connect the pods to the RHOSP network and Octavia is required to create the OpenShift Container Platform services.

### 9.2.10.8. Kuryr ports pools

A Kuryr ports pool maintains a number of ports on standby for pod creation.

Keeping ports on standby minimizes pod creation time. Without ports pools, Kuryr must explicitly request port creation or deletion whenever a pod is created or deleted.

The Neutron ports that Kuryr uses are created in subnets that are tied to namespaces. These pod ports are also added as subports to the primary port of OpenShift Container Platform cluster nodes.

Because Kuryr keeps each namespace in a separate subnet, a separate ports pool is maintained for each namespace-worker pair.

Prior to installing a cluster, you can set the following parameters in the `cluster-network-03-config.yml` manifest file to configure ports pool behavior:

- The `enablePortPoolsPrepopulation` parameter controls pool prepopulation, which forces Kuryr to add ports to the pool when it is created, such as when a new host is added, or a new namespace is created. The default value is `false`.

- The `poolMinPorts` parameter is the minimum number of free ports that are kept in the pool. The default value is `1`.

- The `poolMaxPorts` parameter is the maximum number of free ports that are kept in the pool. A value of `0` disables that upper bound. This is the default setting. If your OpenStack port quota is low, or you have a limited number of IP addresses on the pod network, consider setting this option to ensure that unneeded ports are deleted.

- The `poolBatchPorts` parameter defines the maximum number of Neutron ports that can be created at once. The default value is `3`.

### 9.2.10.9. Adjusting Kuryr ports pools during installation

During installation, you can configure how Kuryr manages Red Hat OpenStack Platform (RHOSP) Neutron ports to control the speed and efficiency of pod creation.

**Prerequisites**

- Create and modify the `install-config.yaml` file.

**Procedure**
1. From a command line, create the manifest files:

   $ ./openshift-install create manifests --dir <installation_directory> ①

   For `<installation_directory>`, specify the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a file that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   $ touch <installation_directory>/manifests/cluster-network-03-config.yml ①

   For `<installation_directory>`, specify the directory name that contains the `manifests/` directory for your cluster.

After creating the file, several network configuration files are in the `manifests/` directory, as shown:

   $ ls <installation_directory>/manifests/cluster-network-*

   Example output

   cluster-network-01-crd.yml
   cluster-network-02-config.yml
   cluster-network-03-config.yml

3. Open the `cluster-network-03-config.yml` file in an editor, and enter a custom resource (CR) that describes the Cluster Network Operator configuration that you want:

   $ oc edit networks.operator.openshift.io cluster

4. Edit the settings to meet your requirements. The following file is provided as an example:

   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     clusterNetwork:
       - cidr: 10.128.0.0/14
         hostPrefix: 23
     serviceNetwork:
       - 172.30.0.0/16
     defaultNetwork:
       type: Kuryr
       kuryrConfig:
         enablePortPoolsPrepopulation: false ①
         poolMinPorts: 1 ②
         poolBatchPorts: 3 ③
         poolMaxPorts: 5 ④
         openstackServiceNetwork: 172.30.0.0/15 ⑤
Set the value of `enablePortPoolsPrepopulation` to `true` to make Kuryr create new Neutron ports after a namespace is created or a new node is added to the cluster. This setting raises the Neutron ports quota but can reduce the time that is required to spawn pods. The default value is `false`.

Kuryr creates new ports for a pool if the number of free ports in that pool is lower than the value of `poolMinPorts`. The default value is `1`.

`poolBatchPorts` controls the number of new ports that are created if the number of free ports is lower than the value of `poolMinPorts`. The default value is `3`.

If the number of free ports in a pool is higher than the value of `poolMaxPorts`, Kuryr deletes them until the number matches that value. Setting this value to `0` disables this upper bound, preventing pools from shrinking. The default value is `0`.

The `openStackServiceNetwork` parameter defines the CIDR range of the network from which IP addresses are allocated to RHOSP Octavia’s LoadBalancers. If this parameter is used with the Amphora driver, Octavia takes two IP addresses from this network for each load balancer: one for OpenShift and the other for VRRP connections. Because these IP addresses are managed by OpenShift Container Platform and Neutron respectively, they must come from different pools. Therefore, the value of `openStackServiceNetwork` must be at least twice the size of the value of `serviceNetwork`, and the value of `serviceNetwork` must overlap entirely with the range that is defined by `openStackServiceNetwork`.

The CNO verifies that VRRP IP addresses that are taken from the range that is defined by this parameter do not overlap with the range that is defined by the `serviceNetwork` parameter.

If this parameter is not set, the CNO uses an expanded value of `serviceNetwork` that is determined by decrementing the prefix size by 1.

Save the `cluster-network-03-config.yml` file, and exit the text editor.

Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory while creating the cluster.

### 9.2.11. Setting compute machine affinity

Optionally, you can set the affinity policy for compute machines during installation. The installer does not select an affinity policy for compute machines by default.

You can also create machine sets that use particular RHOSP server groups after installation.

**NOTE**

Control plane machines are created with a `soft-anti-affinity` policy.

**TIP**

You can learn more about RHOSP instance scheduling and placement in the RHOSP documentation.

**Prerequisites**

- Create the `install-config.yaml` file and complete any modifications to it.
Procedure

1. Using the RHOSP command-line interface, create a server group for your compute machines. For example:

   ```bash
   $ openstack \
   --os-compute-api-version=2.15 \
   server group create \
   --policy anti-affinity \
   my-openshift-worker-group
   ```

   For more information, see the `server group create` command documentation.

2. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir=<installation_directory>
   ```

   where:

   - **installation_directory**
     - Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

3. Open `manifests/99_openshift-cluster-api_worker-machineset-0.yaml`, the MachineSet definition file.

4. Add the property `serverGroupID` to the definition beneath the `spec.template.spec.providerSpec.value` property. For example:

   ```yaml
   apiVersion: machine.openshift.io/v1beta1
   kind: MachineSet
   metadata:
     labels:
       machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
       machine.openshift.io/cluster-api-machine-role: <node_role>
       machine.openshift.io/cluster-api-machine-type: <node_role>
     name: <infrastructure_ID>-<node_role>
     namespace: openshift-machine-api
   spec:
     replicas: <number_of_replicas>
     selector:
       matchLabels:
         machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
         machine.openshift.io/cluster-api-machine-role: <node_role>
         machine.openshift.io/cluster-api-machine-type: <node_role>
     template:
       metadata:
         labels:
           machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
           machine.openshift.io/cluster-api-machine-role: <node_role>
           machine.openshift.io/cluster-api-machine-type: <node_role>
         name: <infrastructure_ID>-<node_role>
       spec:
         providerSpec:
           value:
             apiVersion: openstackproviderconfig.openshift.io/v1alpha1
             cloudName: openstack
   ```
Add the UUID of your server group here.

5. Optional: Back up the `manifests/99_openshift-cluster-api_worker-machineset-0.yaml` file. The installation program deletes the `manifests/` directory when creating the cluster.

When you install the cluster, the installer uses the MachineSet definition that you modified to create compute machines within your RHOSP server group.

**9.2.12. Generating an SSH private key and adding it to the agent**

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

![NOTE]

- In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:
Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```
$ ssh-keygen -t ed25519 -N " \n-f <path>/<file_name>
```

**Example output**

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name>
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**9.2.13. Enabling access to the environment**

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.
You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

### 9.2.13.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

**Procedure**

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

   ```bash
   $ openstack floating ip create --description "API <cluster_name>.<base_domain>" <external_network>
   ```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

   ```bash
   $ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>" <external_network>
   ```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

   ```
   api.<cluster_name>.<base_domain>. IN A <API_FIP>
   *.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
   ```

   **NOTE**

   If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

4. Add the FIPs to the `install-config.yaml` file as the values of the following parameters:

   - `platform.openstack.ingressFloatingIP`
   - `platform.openstack.lbFloatingIP`

   If you use these values, you must also enter an external network as the value of the `platform.openstack.externalNetwork` parameter in the `install-config.yaml` file.

**TIP**

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

### 9.2.13.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `install-config.yaml` file, do not define the following parameters:
• platform.openstack.ingressFloatingIP
• platform.openstack.lbFloatingIP

If you cannot provide an external network, you can also leave platform.openstack.externalNetwork blank. If you do not provide a value for platform.openstack.externalNetwork, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.

**NOTE**
You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

### 9.2.14. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**
You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \  
   --log-level=info  
   ```

   **1** For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`. 
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
INFO Time elapsed: 36m22s
```

NOTE

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

9.2.15. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

**Procedure**
1. In the cluster environment, export the administrator’s kubeconfig file:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig  

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

   $ oc get nodes

3. View your cluster’s version:

   $ oc get clusterversion

4. View your Operators’ status:

   $ oc get clusteroperator

5. View all running pods in the cluster:

   $ oc get pods -A

9.2.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig  

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   $ oc whoami
9.2.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

9.2.18. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If you need to enable external access to node ports, configure ingress cluster traffic by using a node port.
- If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses.

9.3. INSTALLING A CLUSTER ON OPENSTACK ON YOUR OWN INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on Red Hat OpenStack Platform (RHOSP) that runs on user-provisioned infrastructure.

Using your own infrastructure allows you to integrate your cluster with existing infrastructure and modifications. The process requires more labor on your part than installer-provisioned installations, because you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. However, Red Hat provides Ansible playbooks to help you in the deployment process.

9.3.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
  - Verify that OpenShift Container Platform 4.6 is compatible with your RHOSP version in the Available platforms section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
• Verify that your network configuration does not rely on a provider network. Provider networks are not supported.

• Have an RHOSP account where you want to install OpenShift Container Platform.

• On the machine from which you run the installation program, have:
  o A single directory in which you can keep the files you create during the installation process
  o Python 3

### 9.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

• Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

• Access Quay.io to obtain the packages that are required to install your cluster.

• Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 9.3.3. Resource guidelines for installing OpenShift Container Platform on RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

**Table 9.13. Recommended resources for a default OpenShift Container Platform cluster on RHOSP**

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating IP addresses</td>
<td>3</td>
</tr>
<tr>
<td>Ports</td>
<td>15</td>
</tr>
<tr>
<td>Routers</td>
<td>1</td>
</tr>
<tr>
<td>Subnets</td>
<td>1</td>
</tr>
<tr>
<td>RAM</td>
<td>112 GB</td>
</tr>
</tbody>
</table>
### A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

**IMPORTANT**

If RHOSP object storage (Swift) is available and operated by a user account with the `swiftoperator` role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

**NOTE**

By default, your security group and security group rule quotas might be low. If you encounter problems, run `openstack quota set --secgroups 3 --secgroup-rules 60 <project>` as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

#### 9.3.3.1. Control plane and compute machines

By default, the OpenShift Container Platform installation process stands up three control plane and three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

**TIP**

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

#### 9.3.3.2. Bootstrap machine
During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

### 9.3.4. Downloading playbook dependencies

The Ansible playbooks that simplify the installation process on user-provisioned infrastructure require several Python modules. On the machine where you will run the installer, add the modules' repositories and then download them.

**NOTE**

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

#### Prerequisites

- Python 3 is installed on your machine.

#### Procedure

1. On a command line, add the repositories:
   
   a. Register with Red Hat Subscription Manager:
      
      ```bash
      $ sudo subscription-manager register # If not done already
      
      b. Pull the latest subscription data:
      
      ```bash
      $ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
      
      c. Disable the current repositories:
      
      ```bash
      $ sudo subscription-manager repos --disable=* # If not done already
      
      d. Add the required repositories:
      
      ```bash
      $ sudo subscription-manager repos \
      --enable=rhel-8-for-x86_64-baseos-rpms \
      --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
      --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
      --enable=rhel-8-for-x86_64-appstream-rpms
      
      2. Install the modules:
      
      ```bash
      $ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
      ```
3. Ensure that the `python` command points to `python3`:

```
$ sudo alternatives --set python /usr/bin/python3
```

### 9.3.5. Downloading the installation playbooks

Download Ansible playbooks that you can use to install OpenShift Container Platform on your own Red Hat OpenStack Platform (RHOSP) infrastructure.

**Prerequisites**

- The curl command-line tool is available on your machine.

**Procedure**

- To download the playbooks to your working directory, run the following script from a command line:

```
$ xargs -n 1 curl -O <<< 'https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/common.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/inventory.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/download-bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/download-control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/download-security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/download-containers.yaml'
```

The playbooks are downloaded to your machine.
IMPORTANT

During the installation process, you can modify the playbooks to configure your deployment.

Retain all playbooks for the life of your cluster. You must have the playbooks to remove your OpenShift Container Platform cluster from RHOSP.

IMPORTANT

You must match any edits you make in the `bootstrap.yaml`, `compute-nodes.yaml`, `control-plane.yaml`, `network.yaml`, and `security-groups.yaml` files to the corresponding playbooks that are prefixed with `down-`. For example, edits to the `bootstrap.yaml` file must be reflected in the `down-bootstrap.yaml` file, too. If you do not edit both files, the supported cluster removal process will fail.

9.3.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.3.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name> ¹
   ¹ Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.
   
   Running this command generates an SSH key that does not require a password in the location that you specified.
   
   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>  
   ```

   **Example output**

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**9.3.8. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image**

The OpenShift Container Platform installation program requires that a Red Hat Enterprise Linux CoreOS (RHCOS) image be present in the Red Hat OpenStack Platform (RHOSP) cluster. Retrieve the latest RHCOS image, then upload it using the RHOSP CLI.

**Prerequisites**

- The RHOSP CLI is installed.

**Procedure**


2. Under **Version**, select the most recent release of OpenShift Container Platform 4.6 for Red Hat Enterprise Linux (RHEL) 8.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the `Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)`.

4. Decompress the image.
NOTE

You must decompress the RHOSP image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like `.gz` or `.tgz`. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. From the image that you downloaded, create an image that is named `rhcos` in your cluster by using the RHOSP CLI:

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-`{RHCOS_VERSION}`-openstack.qcow2 rhcos
```

IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either `.raw` or `.qcow2` formats. If you use Ceph, you must use the `.raw` format.

WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

After you upload the image to RHOSP, it is usable in the installation process.

9.3.9. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack’s networking service to have DHCP agents forward instances’ DNS queries

Procedure

1. Using the RHOSP CLI, verify the name and ID of the ‘External’ network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+--------------------------------------+----------------+-------------+
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>ID</td>
<td>Router Type</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
```
A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network.

### NOTE

If the Neutron trunk service plug-in is enabled, a trunk port is created by default. For more information, see Neutron trunk port.

#### 9.3.10. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

#### 9.3.10.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API, cluster applications, and the bootstrap process.

### Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

   ```bash
   $ openstack floating ip create --description "API <cluster_name>.<base_domain>" <external_network>
   ```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

   ```bash
   $ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>" <external_network>
   ```

3. By using the Red Hat OpenStack Platform (RHOSP) CLI, create the bootstrap FIP:

   ```bash
   $ openstack floating ip create --description "bootstrap machine" <external_network>
   ```

4. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

   ```
   api.<cluster_name>.<base_domain>. IN A <API_FIP>
   * apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
   ```

---

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Router Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>148a8023-62a7-4672-b018-003462f8d7dc</td>
<td>public_network</td>
<td>External</td>
</tr>
</tbody>
</table>
NOTE
If you do not control the DNS server, you can add the record to your /etc/hosts file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

5. Add the FIPs to the inventory.yaml file as the values of the following variables:
   
   - os_api_fip
   - os_bootstrap_fip
   - os_ingress_fip

If you use these values, you must also enter an external network as the value of the os_external_network variable in the inventory.yaml file.

TIP
You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

9.3.10.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the inventory.yaml file, do not define the following variables:

   - os_api_fip
   - os_bootstrap_fip
   - os_ingress_fip

If you cannot provide an external network, you can also leave os_external_network blank. If you do not provide a value for os_external_network, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. Later in the installation process, when you create network resources, you must configure external connectivity on your own.

If you run the installer with the wait-for command from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.
NOTE

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```plaintext
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

9.3.11. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called `clouds.yaml`. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, login information, and authorization service URLs.

Procedure

1. Create the `clouds.yaml` file:
   - If your RHOSP distribution includes the Horizon web UI, generate a `clouds.yaml` file in it.
     
     **IMPORTANT**
     
     Remember to add a password to the `auth` field. You can also keep secrets in a separate file from `clouds.yaml`.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about `clouds.yaml`, see Config files in the RHOSP documentation.

     ```yaml
     clouds:
     shiftstack:
     auth:
     project_name: shiftstack
     username: shiftstack_user
     password: XXX
     user_domain_name: Default
     project_domain_name: Default
     dev-env:
     region_name: RegionOne
     auth:
     username: 'devuser'
     password: XXX
     project_name: 'devonly'
     ```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
   - Copy the certificate authority file to your machine.
b. Add the machine to the certificate authority trust bundle:

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

c. Update the trust bundle:

```
$ sudo update-ca-trust extract
```

d. Add the `cacerts` key to the `clouds.yaml` file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

**TIP**

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the `ca-cert.pem` key in the `cloud-provider-config` keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the `clouds.yaml` file in one of the following locations:

   a. The value of the `OS_CLIENT_CONFIG_FILE` environment variable

   b. The current directory

   c. A Unix-specific user configuration directory, for example `~/.config/openstack/clouds.yaml`

   d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`

   The installation program searches for `clouds.yaml` in that order.

### 9.3.12. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>
```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `openstack` as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane and compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the [Installation configuration parameters](#) section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

You now have the file `install-config.yaml` in the directory that you specified.

### 9.3.13. Installation configuration parameters
Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 9.3.13.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;</code>, <code>&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>.metadata.name</code> and <code>.baseDomain</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev. The string must be 14 characters or fewer long.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{ &quot;auths&quot;:{ &quot;cloud.openshift.com&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot; }, &quot;quay.io&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot; } } }</td>
</tr>
</tbody>
</table>

9.3.13.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 9.15. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
### networking

**Description:** The configuration for the cluster network.

**Values:** Object

**NOTE:** You cannot modify parameters specified by the `networking` object after installation.

### networking.network Type

**Description:** The cluster network provider Container Network Interface (CNI) plug-in to install.

**Values:** Either `OpenShiftSDN` or `OVNKubernetes`. The default value is `OpenShiftSDN`.

### networking.clusterNetwork

**Description:** The IP address blocks for pods.

**Values:** An array of objects. For example:

```yaml
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
```

### networking.clusterNetwork.cidr

**Description:** Required if you use `networking.clusterNetwork`. An IP address block.

**Values:** An IPv4 network.

### networking.clusterNetwork.hostPrefix

**Description:** The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23 then each node is assigned a /23 subnet out of the given cidr. A `hostPrefix` value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.

**Values:** A subnet prefix. The default value is 23.

### networking.serviceNetwork

**Description:** The IP address block for services. The default value is 172.30.0.0/16.

**Values:** An array with an IP address block in CIDR format. For example:

```yaml
networking:
  serviceNetwork:
  - 172.30.0.0/16
```
### 9.3.13.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>additionalTrustBundle</code></td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td><code>compute</code></td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td><code>compute.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>

#### networking.machinenuNetwork

The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

<table>
<thead>
<tr>
<th>networking.machinenuNetwork</th>
<th>An array of objects. For example: networking: machineNetwork: - cidr: 10.0.0.0/16</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>networking.machinenuNetwork.cidr</th>
<th>Required if you use <code>networking.machinenuNetwork</code>. An IP address block. The default value is <code>10.0.0.0/16</code> for all platforms other than libvirt. For libvirt, the default value is <code>192.168.126.0/24</code>.</th>
</tr>
</thead>
</table>

**NOTE**

Set the `networking.machinenuNetwork` to match the CIDR that the preferred NIC resides in.
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
## controlPlane.hypertreading

Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

### controlPlane.name

Required if you use `controlPlane`. The name of the machine pool.

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>master</td>
</tr>
</tbody>
</table>

### controlPlane.platform

Required if you use `controlPlane`. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the `compute.platform` parameter value.

<table>
<thead>
<tr>
<th>Parameter Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
</tbody>
</table>

### controlPlane.replicas

The number of control plane machines to provision.

<table>
<thead>
<tr>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>The only supported value is 3, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td>dynamically tries to determine the capabilities of the provided credentials,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with a preference for mint mode on the platforms where multiple modes are</td>
<td></td>
</tr>
<tr>
<td></td>
<td>supported.</td>
<td></td>
</tr>
</tbody>
</table>
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fips</code></td>
<td>Sources and repositories for the release-image content.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td><code>imageContentSources</code></td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><code>imageContentSources.source</code></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>String</td>
</tr>
</tbody>
</table>

Array of strings
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**Internal** or **External**. The default value is **External**.

Setting this field to **Internal** is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to **Internal**, the cluster will become non-functional. For more information, refer to [BZ#1953035](#).

**sshKey**

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 9.3.13.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

**Table 9.17. Additional RHOSP parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.rootVolume.size</code></td>
<td>For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example <strong>30</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.openstack.rootVolume.type</code></td>
<td>For compute machines, the root volume’s type.</td>
<td>String, for example <strong>performance</strong>.</td>
</tr>
</tbody>
</table>
For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.

Integer, for example 30.

For control plane machines, the root volume’s type.

String, for example performance.

The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.

String, for example MyCloud.

The RHOSP external network name to be used for installation.

String, for example external.

The RHOSP flavor to use for control plane and compute machines.

String, for example m1.xlarge.

### 9.3.13.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.openstack.aditionalNetworkIDs</td>
<td>Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>compute.platform.openstack.aditionalSecurityGroupIDs</td>
<td>Additional security groups that are associated with compute machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>compute.platform.openstack.zones</strong></td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td><strong>controlPlane.platform.openstack.additionalNetworkIDs</strong></td>
<td>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.openstack.additionalSecurityGroupIDs</strong></td>
<td>Additional security groups that are associated with control plane machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td><strong>controlPlane.platform.openstack.zones</strong></td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>platform.openstack.clusterOSImage</strong></td>
<td>The location from which the installer downloads the RHCOS image.</td>
<td>An HTTP or HTTPS URL, optionally with an SHA-256 checksum. For example, <a href="http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d">http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</a>. The value can also be the name of an existing Glance image, for example my-rhcos.</td>
</tr>
<tr>
<td><strong>platform.openstack.defaultMachinePlatform</strong></td>
<td>The default machine pool platform configuration.</td>
<td>{ &quot;type&quot;: &quot;ml.large&quot;, &quot;rootVolume&quot;: { &quot;size&quot;: 30, &quot;type&quot;: &quot;performance&quot; } }</td>
</tr>
<tr>
<td><strong>platform.openstack.ingressFloatingIP</strong></td>
<td>An existing floating IP address to associate with the Ingress port.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><strong>platform.openstack.lbFloatingIP</strong></td>
<td>An existing floating IP address to associate with the API load balancer.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><strong>platform.openstack.externalDNS</strong></td>
<td>IP addresses for external DNS servers that cluster instances use for DNS resolution.</td>
<td>A list of IP addresses as strings. For example, [&quot;8.8.8.8&quot;, &quot;192.168.1.12&quot;]</td>
</tr>
</tbody>
</table>
### 9.3.13.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet’s GUID is passed as the value of `platform.openstack.machinesSubnet` in the `install-config.yaml` file.

This subnet is used as the cluster’s primary subnet; nodes and ports are created on it.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that:

- The target network and subnet are available.
- DHCP is enabled on the target subnet.
- You can provide installer credentials that have permission to create ports on the target network.
- If your network configuration requires a router, it is created in RHOSP. Some configurations rely on routers for floating IP address translation.
- Your network configuration does not rely on a provider network. Provider networks are not supported.

**NOTE**

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network’s CIDR block. To override these default values, set values for `platform.openstack.apiVIP` and `platform.openstack.ingressVIP` that are outside of the DHCP allocation pool.

### 9.3.13.7. Sample customized install-config.yaml file for RHOSP

This sample `install-config.yaml` demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

---

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.openstack.machinesSubnet</code></td>
<td>The UUID of a RHOSP subnet that the cluster’s nodes use. Nodes and virtual IP (VIP) ports are created on this subnet. The first item in <code>networking.machineNetwork</code> must match the value of <code>machinesSubnet</code>. If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</td>
<td>A UUID as a string. For example, <code>fa806b2f-ac49-4bce-b9db-124bc64209bf</code>.</td>
</tr>
</tbody>
</table>
9.3.13.8. Setting a custom subnet for machines

The IP range that the installation program uses by default might not match the Neutron subnet that you create when you install OpenShift Container Platform. If necessary, update the CIDR value for new machines by editing the installation configuration file.

**Prerequisites**

- You have the `install-config.yaml` file that was generated by the OpenShift Container Platform installation program.

**Procedure**

1. On a command line, browse to the directory that contains `install-config.yaml`.
2. From that directory, either run a script to edit the `install-config.yaml` file or update the file manually:

- To set the value by using a script, run:

  ```
  $ python -c "
  import yaml;
  path = "install-config.yaml";
  data = yaml.safe_load(open(path));
  data["networking"]["machineNetwork"] = ["cidr": "192.168.0.0/18"];
  open(path, "w").write(yaml.dump(data, default_flow_style=False))"
  
  Insert a value that matches your intended Neutron subnet, e.g. `192.0.2.0/24`.

- To set the value manually, open the file and set the value of `networking.machineCIDR` to something that matches your intended Neutron subnet.

9.3.13.9. Emptying compute machine pools

To proceed with an installation that uses your own infrastructure, set the number of compute machines in the installation configuration file to zero. Later, you create these machines manually.

Prerequisites

- You have the `install-config.yaml` file that was generated by the OpenShift Container Platform installation program.

Procedure

1. On a command line, browse to the directory that contains `install-config.yaml`.

2. From that directory, either run a script to edit the `install-config.yaml` file or update the file manually:

- To set the value by using a script, run:

  ```
  $ python -c "
  import yaml;
  path = "install-config.yaml";
  data = yaml.safe_load(open(path));
  data["compute"][0]["replicas"] = 0;
  open(path, "w").write(yaml.dump(data, default_flow_style=False))"
  
  To set the value manually, open the file and set the value of `compute.<first entry>.replicas` to 0.

9.3.14. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the `install-config.yaml` installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Example output

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.

   - You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Example output

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.
Open! the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```markdown
   ├── auth
   │    ├── kubeadm-password
   │    └── kubecfg
   │    └── bootstrap.ign
   ├── master.ign
   ├── metadata.json
   └── worker.ign
   ```

5. Export the metadata file’s `infraID` key as an environment variable:

   ```bash
   export INFRA_ID=$(jq -r .infraID metadata.json)
   ```

   **TIP**

   Extract the `infraID` key from `metadata.json` and use it as a prefix for all of the RHOSP resources that you create. By doing so, you avoid name conflicts when making multiple deployments in the same project.

### 9.3.15. Preparing the bootstrap Ignition files

The OpenShift Container Platform installation process relies on bootstrap machines that are created from a bootstrap Ignition configuration file.

Edit the file and upload it. Then, create a secondary bootstrap Ignition configuration file that Red Hat OpenStack Platform (RHOSP) uses to download the primary file.

**Prerequisites**

- You have the bootstrap Ignition file that the installer program generates, `bootstrap.ign`.

- The infrastructure ID from the installer’s metadata file is set as an environment variable (`$INFRA_ID`).
  - If the variable is not set, see Creating the Kubernetes manifest and Ignition config files

- You have an HTTP(S)-accessible way to store the bootstrap Ignition file.
  - The documented procedure uses the RHOSP image service (Glance), but you can also use the RHOSP storage service (Swift), Amazon S3, an internal HTTP server, or an ad hoc Nova
Procedure

1. Run the following Python script. The script modifies the bootstrap Ignition file to set the hostname and, if available, CA certificate file when it runs:

```python
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap
').decode().strip()
files.append(
    {
        'path': '/etc/hostname',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64, ' + hostname_b64
        }
    })

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
    ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()
    files.append(
        {
            'path': '/opt/openshift/tls/cloud-ca-cert.pem',
            'mode': 420,
            'contents': {
                'source': 'data:text/plain;charset=utf-8;base64, ' + ca_cert_b64
            }
        })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. Using the RHOSP CLI, create an image that uses the bootstrap Ignition file:

```
$ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign <image_name>
```

3. Get the image’s details:

```
$ openstack image show <image_name>
```
Make a note of the file value; it follows the pattern `v2/images/<image_ID>/file`.

**NOTE**
Verify that the image you created is active.

4. Retrieve the image service’s public address:

   ```bash
   $ openstack catalog show image
   ```

5. Combine the public address with the image file value and save the result as the storage location. The location follows the pattern `<image_service_public_URL>/v2/images/<image_ID>/file`.

6. Generate an auth token and save the token ID:

   ```bash
   $ openstack token issue -c id -f value
   ```

7. Insert the following content into a file called `$INFRA_ID-bootstrap-ignition.json` and edit the placeholders to match your own values:

   ```json
   {
   "ignition": {
      "config": {
         "merge": [ {
            "source": "<storage_url>",
            "httpHeaders": [ {  
              "name": "X-Auth-Token",
              "value": "<token_ID>"
            } ]
         } ],
         "security": {  
            "tls": {  
              "certificateAuthorities": [ {  
                "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>"
              } ]
            }  
        },
        "version": "3.1.0"
      }
   }
   ```

   **1** Replace the value of `ignition.config.merge.source` with the bootstrap Ignition file storage URL.

   **2** Set `name` in `httpHeaders` to "X-Auth-Token".

   **3** Set `value` in `httpHeaders` to your token’s ID.

   **4** If the bootstrap Ignition file server uses a self-signed certificate, include the base64-encoded certificate.
8. Save the secondary Ignition config file.

The bootstrap Ignition data will be passed to RHOSP during installation.

**WARNING**

The bootstrap Ignition file contains sensitive information, like `clouds.yaml` credentials. Ensure that you store it in a secure place, and delete it after you complete the installation process.

9.3.16. Creating control plane Ignition config files on RHOSP

Installing OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) on your own infrastructure requires control plane Ignition config files. You must create multiple config files.

**NOTE**

As with the bootstrap Ignition configuration, you must explicitly define a hostname for each control plane machine.

**Prerequisites**

- The infrastructure ID from the installation program’s metadata file is set as an environment variable (`$INFRA_ID`).
  - If the variable is not set, see "Creating the Kubernetes manifest and Ignition config files."

**Procedure**

- On a command line, run the following Python script:

  ```
  $ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index"
    python -c "import base64, json, sys;
    ignition = json.load(sys.stdin);
    storage = ignition.get('storage', {});
    files = storage.get('files', []);
    files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
        'data:text/plain;charset=utf-8;base64,' +
        base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(),
        'verification': {}}},
    'filesystem': 'root'});
    storage['files'] = files;
    files = storage.get('files', []);
    files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
        'data:text/plain;charset=utf-8;base64,' +
        base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(),
        'verification': {}}},
    'filesystem': 'root'});
    storage['files'] = files;
    files = storage.get('files', []);
    files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
        'data:text/plain;charset=utf-8;base64,' +
        base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(),
        'verification': {}}},
    'filesystem': 'root'});
    json.dump(ignition, sys.stdout)
  done
  ```

You now have three control plane Ignition files: `<INFRA_ID>-master-0-ignition.json`, `<INFRA_ID>-master-1-ignition.json`, and `<INFRA_ID>-master-2-ignition.json`.

9.3.17. Creating network resources on RHOSP
Create the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires. To save time, run supplied Ansible playbooks that generate security groups, networks, subnets, routers, and ports.

**Prerequisites**

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."

**Procedure**

1. Optional: Add an external network value to the `inventory.yaml` playbook:

   **Example external network value in the inventory.yaml Ansible playbook**

   ```yaml
   ...  
   # The public network providing connectivity to the cluster. If not
   # provided, the cluster external connectivity must be provided in another
   # way.
   
   # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
   os_external_network: 'external'
   ...
   
   IMPORTANT
   
   If you did not provide a value for `os_external_network` in the `inventory.yaml` file, you must ensure that VMs can access Glance and an external connection yourself.

2. Optional: Add external network and floating IP (FIP) address values to the `inventory.yaml` playbook:

   **Example FIP values in the inventory.yaml Ansible playbook**

   ```yaml
   ...
   # OpenShift API floating IP address. If this value is non-empty, the
   # corresponding floating IP will be attached to the Control Plane to
   # serve the OpenShift API.
   os_api_fip: '203.0.113.23'
   
   # OpenShift Ingress floating IP address. If this value is non-empty, the
   # corresponding floating IP will be attached to the worker nodes to serve
   # the applications.
   os_ingress_fip: '203.0.113.19'
   
   # If this value is non-empty, the corresponding floating IP will be
   # attached to the bootstrap machine. This is needed for collecting logs
   # in case of install failure.
   os_bootstrap_fip: '203.0.113.20'
   ...
   ```
IMPORTANT

If you do not define values for `os_api_fip` and `os_ingress_fip`, you must perform post-installation network configuration.

If you do not define a value for `os_bootstrap_fip`, the installer cannot download debugging information from failed installations.

See "Enabling access to the environment" for more information.

3. On a command line, create security groups by running the `security-groups.yaml` playbook:

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. On a command line, create a network, subnet, and router by running the `network.yaml` playbook:

```
$ ansible-playbook -i inventory.yaml network.yaml
```

5. Optional: If you want to control the default resolvers that Nova servers use, run the RHOSP CLI command:

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2> "$INFRA_ID-nodes"
```

9.3.18. Creating the bootstrap machine on RHOSP

Create a bootstrap machine and give it the network access it needs to run on Red Hat OpenStack Platform (RHOSP). Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The `inventory.yaml`, `common.yaml`, and `bootstrap.yaml` Ansible playbooks are in a common directory.
- The `metadata.json` file that the installation program created is in the same directory as the Ansible playbooks.

Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. On a command line, run the `bootstrap.yaml` playbook:

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. After the bootstrap server is active, view the logs to verify that the Ignition files were received:

```
$ openstack console log show "$INFRA_ID-bootstrap"
```
9.3.19. Creating the control plane machines on RHOSP

Create three control plane machines by using the Ignition config files that you generated. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The infrastructure ID from the installation program’s metadata file is set as an environment variable ($INFRA_ID).
- The `inventory.yaml`, `common.yaml`, and `control-plane.yaml` Ansible playbooks are in a common directory.
- You have the three Ignition files that were created in "Creating control plane Ignition config files."

Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. If the control plane Ignition config files aren’t already in your working directory, copy them into it.

3. On a command line, run the `control-plane.yaml` playbook:

   ```bash
   $ ansible-playbook -i inventory.yaml control-plane.yaml
   ```

4. Run the following command to monitor the bootstrapping process:

   ```bash
   $ openshift-install wait-for bootstrap-complete
   ```

   You will see messages that confirm that the control plane machines are running and have joined the cluster:

   ```
   INFO API v1.14.6+f9b5405 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   ...
   INFO It is now safe to remove the bootstrap resources
   ```

9.3.20. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
• You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   1 For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   $ oc whoami

   Example output

   system:admin

9.3.21. Deleting bootstrap resources from RHOSP

Delete the bootstrap resources that you no longer need.

Prerequisites

• You downloaded the modules in "Downloading playbook dependencies."

• You downloaded the playbooks in "Downloading the installation playbooks."

• The inventory.yaml, common.yaml, and down-bootstrap.yaml Ansible playbooks are in a common directory.

• The control plane machines are running.
  • If you do not know the status of the machines, see "Verifying cluster status."

Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. On a command line, run the down-bootstrap.yaml playbook:

   $ ansible-playbook -i inventory.yaml down-bootstrap.yaml

   The bootstrap port, server, and floating IP address are deleted.
WARNING
If you did not disable the bootstrap Ignition file URL earlier, do so now.

9.3.22. Creating compute machines on RHOSP

After standing up the control plane, create compute machines. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The `inventory.yaml`, `common.yaml`, and `compute-nodes.yaml` Ansible playbooks are in a common directory.
- The `metadata.json` file that the installation program created is in the same directory as the Ansible playbooks.
- The control plane is active.

Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. On a command line, run the playbook:

   ```
   $ ansible-playbook -i inventory.yaml compute-nodes.yaml
   ```

Next steps

- Approve the certificate signing requests for the machines.

9.3.23. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   ```
Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME        AGE     REQUESTOR                                      CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.
NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name> 1
  
  1  <csr_name> is the name of a CSR from the list of current CSRs.
  ```

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs --no-run-if-empty oc adm certificate approve
  ```

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name> 1
  
  1  <csr_name> is the name of a CSR from the list of current CSRs.
  ```
To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```bash
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

### 9.3.24. Verifying a successful installation

Verify that the OpenShift Container Platform installation is complete.

**Prerequisites**

- You have the installation program (**openshift-install**)

**Procedure**

- On a command line, enter:

  ```bash
  $ openshift-install --log-level debug wait-for install-complete
  ```

The program outputs the console URL, as well as the administrator’s login information.

### 9.3.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**
- See About remote health monitoring for more information about the Telemetry service

**9.3.26. Next steps**
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If you need to enable external access to node ports, configure ingress cluster traffic by using a node port.
- If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses.

**9.4. INSTALLING A CLUSTER ON OPENSTACK WITH KURYR ON YOUR OWN INFRASTRUCTURE**

In OpenShift Container Platform version 4.6, you can install a cluster on Red Hat OpenStack Platform (RHOSP) that runs on user-provisioned infrastructure.

Using your own infrastructure allows you to integrate your cluster with existing infrastructure and modifications. The process requires more labor on your part than installer-provisioned installations, because you must create all RHOSP resources, like Nova servers, Neutron ports, and security groups. However, Red Hat provides Ansible playbooks to help you in the deployment process.

**9.4.1. Prerequisites**
- Review details about the OpenShift Container Platform installation and update processes.
  - Verify that OpenShift Container Platform 4.6 is compatible with your RHOSP version in the Available platforms section. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
- Verify that your network configuration does not rely on a provider network. Provider networks are not supported.
- Have an RHOSP account where you want to install OpenShift Container Platform.
- On the machine from which you run the installation program, have:
  - A single directory in which you can keep the files you create during the installation process
  - Python 3

**9.4.2. About Kuryr SDN**

Kuryr is a container network interface (CNI) plug-in solution that uses the Neutron and Octavia Red Hat OpenStack Platform (RHOSP) services to provide networking for pods and Services.
Kuryr and OpenShift Container Platform integration is primarily designed for OpenShift Container Platform clusters running on RHOSP VMs. Kuryr improves the network performance by plugging OpenShift Container Platform pods into RHOSP SDN. In addition, it provides interconnectivity between pods and RHOSP virtual instances.

Kuryr components are installed as pods in OpenShift Container Platform using the `openshift-kuryr` namespace:

- **kuryr-controller** - a single service instance installed on a **master** node. This is modeled in OpenShift Container Platform as a **Deployment** object.

- **kuryr-cni** - a container installing and configuring Kuryr as a CNI driver on each OpenShift Container Platform node. This is modeled in OpenShift Container Platform as a **DaemonSet** object.

The Kuryr controller watches the OpenShift Container Platform API server for pod, service, and namespace create, update, and delete events. It maps the OpenShift Container Platform API calls to corresponding objects in Neutron and Octavia. This means that every network solution that implements the Neutron trunk port functionality can be used to back OpenShift Container Platform via Kuryr. This includes open source solutions such as Open vSwitch (OVS) and Open Virtual Network (OVN) as well as Neutron-compatible commercial SDNs.

Kuryr is recommended for OpenShift Container Platform deployments on encapsulated RHOSP tenant networks to avoid double encapsulation, such as running an encapsulated OpenShift Container Platform SDN over an RHOSP network.

If you use provider networks or tenant VLANs, you do not need to use Kuryr to avoid double encapsulation. The performance benefit is negligible. Depending on your configuration, though, using Kuryr to avoid having two overlays might still be beneficial.

Kuryr is not recommended in deployments where all of the following criteria are true:

- The RHOSP version is less than 16.

- The deployment uses UDP services, or a large number of TCP services on few hypervisors.

or

- The **ovn-octavia** Octavia driver is disabled.

- The deployment uses a large number of TCP services on few hypervisors.

### 9.4.3. Resource guidelines for installing OpenShift Container Platform on RHOSP with Kuryr

When using Kuryr SDN, the pods, services, namespaces, and network policies are using resources from the RHOSP quota; this increases the minimum requirements. Kuryr also has some additional requirements on top of what a default install requires.

Use the following quota to satisfy a default cluster’s minimum requirements:

**Table 9.19. Recommended resources for a default OpenShift Container Platform cluster on RHOSP with Kuryr**
<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating IP addresses</td>
<td>3 - plus the expected number of Services of LoadBalancer type</td>
</tr>
<tr>
<td>Ports</td>
<td>1500 - 1 needed per Pod</td>
</tr>
<tr>
<td>Routers</td>
<td>1</td>
</tr>
<tr>
<td>Subnets</td>
<td>250 - 1 needed per Namespace/Project</td>
</tr>
<tr>
<td>Networks</td>
<td>250 - 1 needed per Namespace/Project</td>
</tr>
<tr>
<td>RAM</td>
<td>112 GB</td>
</tr>
<tr>
<td>vCPUs</td>
<td>28</td>
</tr>
<tr>
<td>Volume storage</td>
<td>275 GB</td>
</tr>
<tr>
<td>Instances</td>
<td>7</td>
</tr>
<tr>
<td>Security groups</td>
<td>250 - 1 needed per Service and per NetworkPolicy</td>
</tr>
<tr>
<td>Security group rules</td>
<td>1000</td>
</tr>
<tr>
<td>Load balancers</td>
<td>100 - 1 needed per Service</td>
</tr>
<tr>
<td>Load balancer listeners</td>
<td>500 - 1 needed per Service-exposed port</td>
</tr>
<tr>
<td>Load balancer pools</td>
<td>500 - 1 needed per Service-exposed port</td>
</tr>
</tbody>
</table>

A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

**IMPORTANT**

If RHOSP object storage (Swift) is available and operated by a user account with the `swiftoperator` role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

**IMPORTANT**

If you are using Red Hat OpenStack Platform (RHOSP) version 16 with the Amphora driver rather than the OVN Octavia driver, security groups are associated with service accounts instead of user projects.

Take the following notes into consideration when setting resources:
The number of ports that are required is larger than the number of pods. Kuryr uses ports pools to have pre-created ports ready to be used by pods and speed up the pods’ booting time.

Each network policy is mapped into an RHOSP security group, and depending on the NetworkPolicy spec, one or more rules are added to the security group.

Each service is mapped to an RHOSP load balancer. Consider this requirement when estimating the number of security groups required for the quota. If you are using RHOSP version 15 or earlier, or the ovn-octavia driver, each load balancer has a security group with the user project.

The quota does not account for load balancer resources (such as VM resources), but you must consider these resources when you decide the RHOSP deployment’s size. The default installation will have more than 50 load balancers; the clusters must be able to accommodate them. If you are using RHOSP version 16 with the OVN Octavia driver enabled, only one load balancer VM is generated; services are load balanced through OVN flows.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

To enable Kuryr SDN, your environment must meet the following requirements:

- Run RHOSP 13+.
- Have Overcloud with Octavia.
- Use Neutron Trunk ports extension.
- Use openvswitch firewall driver if ML2/OVS Neutron driver is used instead of ovs-hybrid.

9.4.3.1. Increasing quota

When using Kuryr SDN, you must increase quotas to satisfy the Red Hat OpenStack Platform (RHOSP) resources used by pods, services, namespaces, and network policies.

Procedure

- Increase the quotas for a project by running the following command:

  $ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 --ports 1500 --subnets 250 --networks 250 <project>

9.4.3.2. Configuring Neutron

Kuryr CNI leverages the Neutron Trunks extension to plug containers into the Red Hat OpenStack Platform (RHOSP) SDN, so you must use the trunks extension for Kuryr to properly work.

In addition, if you leverage the default ML2/OVS Neutron driver, the firewall must be set to openvswitch instead of ovs_hybrid so that security groups are enforced on trunk subports and Kuryr can properly handle network policies.

9.4.3.3. Configuring Octavia

If you are using Red Hat OpenStack Platform (RHOSP) Octavia, you must use the Octavia extension.
Kuryr SDN uses Red Hat OpenStack Platform (RHOSP)’s Octavia LBaaS to implement OpenShift Container Platform services. Thus, you must install and configure Octavia components in RHOSP to use Kuryr SDN.

To enable Octavia, you must include the Octavia service during the installation of the RHOSP Overcloud, or upgrade the Octavia service if the Overcloud already exists. The following steps for enabling Octavia apply to both a clean install of the Overcloud or an Overcloud update.

**NOTE**

The following steps only capture the key pieces required during the deployment of RHOSP when dealing with Octavia. It is also important to note that registry methods vary.

This example uses the local registry method.

**Procedure**

1. If you are using the local registry, create a template to upload the images to the registry. For example:

   ```bash
   (undercloud) $ openstack overcloud container image prepare \
   -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
   --namespace=registry.access.redhat.com/rhosp13 \
   --push-destination=<local-ip-from-undercloud.conf>:8787 \
   --prefix=openstack- \
   --tag-from-label {version}-{release} \
   --output-env-file=/home/stack/templates/overcloud_images.yaml \
   --output-images-file /home/stack/local_registry_images.yaml
   ```

2. Verify that the `local_registry_images.yaml` file contains the Octavia images. For example:

   ```yaml
   ...
   - imagnename: registry.access.redhat.com/rhosp13/openstack-octavia-api:13.0-43
     push_destination: <local-ip-from-undercloud.conf>:8787
   - imagnename: registry.access.redhat.com/rhosp13/openstack-octavia-health-manager:13.0-45
     push_destination: <local-ip-from-undercloud.conf>:8787
   - imagnename: registry.access.redhat.com/rhosp13/openstack-octavia-housekeeping:13.0-45
     push_destination: <local-ip-from-undercloud.conf>:8787
   - imagnename: registry.access.redhat.com/rhosp13/openstack-octavia-worker:13.0-44
     push_destination: <local-ip-from-undercloud.conf>:8787
   ```

   **NOTE**

   The Octavia container versions vary depending upon the specific RHOSP release installed.

3. Pull the container images from `registry.redhat.io` to the Undercloud node:

   ```bash
   (undercloud) $ sudo openstack overcloud container image upload \
   --config-file /home/stack/local_registry_images.yaml \
   --verbose
   ```

   This may take some time depending on the speed of your network and Undercloud disk.
4. Since an Octavia load balancer is used to access the OpenShift Container Platform API, you must increase their listeners’ default timeouts for the connections. The default timeout is 50 seconds. Increase the timeout to 20 minutes by passing the following file to the Overcloud deploy command:

```
(undercloud) $ cat octavia_timeouts.yaml
parameter_defaults:
  OctaviaTimeoutClientData: 1200000
  OctaviaTimeoutMemberData: 1200000
```

**NOTE**

This is not needed for RHOSP 13.0.13+.

5. Install or update your Overcloud environment with Octavia:

```
$ openstack overcloud deploy --templates \
- e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml \
- e octavia_timeouts.yaml
```

**NOTE**

This command only includes the files associated with Octavia; it varies based on your specific installation of RHOSP. See the RHOSP documentation for further information. For more information on customizing your Octavia installation, see installation of Octavia using Director.

**NOTE**

When leveraging Kuryr SDN, the Overcloud installation requires the Neutron trunk extension. This is available by default on director deployments. Use the opensvswitch firewall instead of the default ovs-hybrid when the Neutron backend is ML2/OVS. There is no need for modifications if the backend is ML2/OVN.

6. In RHOSP versions earlier than 13.0.13, add the project ID to the octavia.conf configuration file after you create the project.

- To enforce network policies across services, like when traffic goes through the Octavia load balancer, you must ensure Octavia creates the Amphora VM security groups on the user project. This change ensures that required load balancer security groups belong to that project, and that they can be updated to enforce services isolation.

**NOTE**

This task is unnecessary in RHOSP version 13.0.13 or later.

Octavia implements a new ACL API that restricts access to the load balancers VIP.

a. Get the project ID
b. Add the project ID to `octavia.conf` for the controllers.
   
i. Source the `stackrc` file:

   ```
   $ source stackrc  # Undercloud credentials
   ```

   ii. List the Overcloud controllers:

   ```
   $ openstack server list
   ```

   Example output

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Status</th>
<th>Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>6bef8e73-2ba5-4860-a0b1-3937f8ca7e01</td>
<td>controller-0</td>
<td>ACTIVE</td>
<td>ctlplane=192.168.24.8, overcloud-full, controller</td>
</tr>
<tr>
<td>dda3173a-ab26-47f8-a2dc-8473b4a67ab9</td>
<td>compute-0</td>
<td>ACTIVE</td>
<td>ctlplane=192.168.24.6, overcloud-full, compute</td>
</tr>
</tbody>
</table>

   iii. SSH into the controller(s).

   ```
   $ ssh heat-admin@192.168.24.8
   ```

   iv. Edit the `octavia.conf` file to add the project into the list of projects where Amphora security groups are on the user’s account.
# List of project IDs that are allowed to have Load balancer security groups belonging to them.

```
amp_secgroup_allowed_projects = PROJECT_ID
```

c. Restart the Octavia worker so the new configuration loads.

```
c. controller-0$ sudo docker restart octavia_worker
```

NOTE

Depending on your RHOSP environment, Octavia might not support UDP listeners. If you use Kuryr SDN on RHOSP version 13.0.13 or earlier, UDP services are not supported. RHOSP version 16 or later support UDP.

9.4.3.3.1. The Octavia OVN Driver

Octavia supports multiple provider drivers through the Octavia API.

To see all available Octavia provider drivers, on a command line, enter:

```
$ openstack loadbalancer provider list
```

Example output

```
+---------+-------------------------------------------------+
| name    | description                                     |
+---------+-------------------------------------------------+
| amphora | The Octavia Amphora driver.                     |
| octavia | Deprecated alias of the Octavia Amphora driver. |
| ovn     | Octavia OVN driver.                             |
+---------+-------------------------------------------------+
```

Beginning with RHOSP version 16, the Octavia OVN provider driver (ovn) is supported on OpenShift Container Platform on RHOSP deployments.

**ovn** is an integration driver for the load balancing that Octavia and OVN provide. It supports basic load balancing capabilities, and is based on OpenFlow rules. The driver is automatically enabled in Octavia by Director on deployments that use OVN Neutron ML2.

The Amphora provider driver is the default driver. If **ovn** is enabled, however, Kuryr uses it.

If Kuryr uses **ovn** instead of Amphora, it offers the following benefits:

- Decreased resource requirements. Kuryr does not require a load balancer VM for each service.
- Reduced network latency.
- Increased service creation speed by using OpenFlow rules instead of a VM for each service.
- Distributed load balancing actions across all nodes instead of centralized on Amphora VMs.

9.4.3.4. Known limitations of installing with Kuryr

Using OpenShift Container Platform with Kuryr SDN has several known limitations.
**RHOSP general limitations**

OpenShift Container Platform with Kuryr SDN does not support `Service` objects with type `NodePort`.

If the machines subnet is not connected to a router, or if the subnet is connected, but the router has no external gateway set, Kuryr cannot create floating IPs for `Service` objects with type `LoadBalancer`.

- Configuring the `sessionAffinity=ClientIP` property on `Service` objects does not have an effect. Kuryr does not support this setting.

**RHOSP version limitations**

Using OpenShift Container Platform with Kuryr SDN has several limitations that depend on the RHOSP version.

- RHOSP versions before 16 use the default Octavia load balancer driver (Amphora). This driver requires that one Amphora load balancer VM is deployed per OpenShift Container Platform service. Creating too many services can cause you to run out of resources. Deployments of later versions of RHOSP that have the OVN Octavia driver disabled also use the Amphora driver. They are subject to the same resource concerns as earlier versions of RHOSP.

- Octavia RHOSP versions before 13.0.13 do not support UDP listeners. Therefore, OpenShift Container Platform UDP services are not supported.

- Octavia RHOSP versions before 13.0.13 cannot listen to multiple protocols on the same port. Services that expose the same port to different protocols, like TCP and UDP, are not supported.

- Kuryr SDN does not support automatic unidling by a service.

**RHOSP environment limitations**

There are limitations when using Kuryr SDN that depend on your deployment environment.

Because of Octavia’s lack of support for the UDP protocol and multiple listeners, if the RHOSP version is earlier than 13.0.13, Kuryr forces pods to use TCP for DNS resolution.

In Go versions 1.12 and earlier, applications that are compiled with CGO support disabled use UDP only. In this case, the native Go resolver does not recognize the `use-vc` option in `resolv.conf`, which controls whether TCP is forced for DNS resolution. As a result, UDP is still used for DNS resolution, which fails.

To ensure that TCP forcing is allowed, compile applications either with the environment variable `CGO_ENABLED` set to 1, i.e. `CGO_ENABLED=1`, or ensure that the variable is absent.

In Go versions 1.13 and later, TCP is used automatically if DNS resolution using UDP fails.

**NOTE**

musl-based containers, including Alpine-based containers, do not support the `use-vc` option.

**RHOSP upgrade limitations**

As a result of the RHOSP upgrade process, the Octavia API might be changed, and upgrades to the Amphora images that are used for load balancers might be required.

You can address API changes on an individual basis.
If the Amphora image is upgraded, the RHOSP operator can handle existing load balancer VMs in two ways:

- Upgrade each VM by triggering a load balancer failover.
- Leave responsibility for upgrading the VMs to users.

If the operator takes the first option, there might be short downtimes during failovers.

If the operator takes the second option, the existing load balancers will not support upgraded Octavia API features, like UDP listeners. In this case, users must recreate their Services to use these features.

**IMPORTANT**

If OpenShift Container Platform detects a new Octavia version that supports UDP load balancing, it recreates the DNS service automatically. The service recreation ensures that the service default supports UDP load balancing.

The recreation causes the DNS service approximately one minute of downtime.

### 9.4.3.5. Control plane and compute machines

By default, the OpenShift Container Platform installation process stands up three control plane and three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

**TIP**

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

### 9.4.3.6. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

### 9.4.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:
- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 9.4.5. Downloading playbook dependencies

The Ansible playbooks that simplify the installation process on user-provisioned infrastructure require several Python modules. On the machine where you will run the installer, add the modules’ repositories and then download them.

**NOTE**

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

**Prerequisites**

- Python 3 is installed on your machine.

**Procedure**

1. On a command line, add the repositories:
   
   a. Register with Red Hat Subscription Manager:
      
      ```bash
      $ sudo subscription-manager register # If not done already
      ```
   
   b. Pull the latest subscription data:
      
      ```bash
      $ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
      ```
   
   c. Disable the current repositories:
      
      ```bash
      $ sudo subscription-manager repos --disable=* # If not done already
      ```
   
   d. Add the required repositories:
      
      ```bash
      $ sudo subscription-manager repos \
      --enable=rhel-8-for-x86_64-baseos-rpms \
      --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
      ```
2. Install the modules:

```bash
$ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-netaddr
```

3. Ensure that the `python` command points to `python3`:

```bash
$ sudo alternatives --set python /usr/bin/python3
```

### 9.4.6. Downloading the installation playbooks

Download Ansible playbooks that you can use to install OpenShift Container Platform on your own Red Hat OpenStack Platform (RHOSP) infrastructure.

#### Prerequisites

- The `curl` command-line tool is available on your machine.

#### Procedure

- To download the playbooks to your working directory, run the following script from a command line:

```bash
$ xargs -n 1 curl -O <<< 'https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/common.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/inventory.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-bootstrap.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-compute-nodes.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-control-plane.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-network.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-security-groups.yaml
https://raw.githubusercontent.com/openshift/installer/release-4.6/upi/openstack/down-containers.yaml'
```
The playbooks are downloaded to your machine.

**IMPORTANT**

During the installation process, you can modify the playbooks to configure your deployment.

Retain all playbooks for the life of your cluster. You must have the playbooks to remove your OpenShift Container Platform cluster from RHOSP.

**IMPORTANT**

You must match any edits you make in the `bootstrap.yaml`, `compute-nodes.yaml`, `control-plane.yaml`, `network.yaml`, and `security-groups.yaml` files to the corresponding playbooks that are prefixed with `down-`. For example, edits to the `bootstrap.yaml` file must be reflected in the `down-bootstrap.yaml` file, too. If you do not edit both files, the supported cluster removal process will fail.

### 9.4.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

9.4.8. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   Agent pid 31874
NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name>
```

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

9.4.9. Creating the Red Hat Enterprise Linux CoreOS (RHCOS) image

The OpenShift Container Platform installation program requires that a Red Hat Enterprise Linux CoreOS (RHCOS) image be present in the Red Hat OpenStack Platform (RHOSP) cluster. Retrieve the latest RHCOS image, then upload it using the RHOSP CLI.

Prerequisites

- The RHOSP CLI is installed.

Procedure


2. Under Version, select the most recent release of OpenShift Container Platform 4.6 for Red Hat Enterprise Linux (RHEL) 8.

   IMPORTANT
   
   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the `Red Hat Enterprise Linux CoreOS (RHCOS) - OpenStack Image (QCOW)`.

4. Decompress the image.
NOTE

You must decompress the RHOSP image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like `.gz` or `.tgz`. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. From the image that you downloaded, create an image that is named `rhcos` in your cluster by using the RHOSP CLI:

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-$(RHCOS_VERSION)-openstack.qcow2 rhcos
```

IMPORTANT

Depending on your RHOSP environment, you might be able to upload the image in either `.raw` or `.qcow2` formats. If you use Ceph, you must use the `.raw` format.

WARNING

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

After you upload the image to RHOSP, it is usable in the installation process.

9.4.10. Verifying external network access

The OpenShift Container Platform installation process requires external network access. You must provide an external network value to it, or deployment fails. Before you begin the process, verify that a network with the external router type exists in Red Hat OpenStack Platform (RHOSP).

Prerequisites

- Configure OpenStack’s networking service to have DHCP agents forward instances’ DNS queries

Procedure

1. Using the RHOSP CLI, verify the name and ID of the ‘External’ network:

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

Example output

```
+--------------------------------------+----------------+-------------+
```
A network with an external router type appears in the network list. If at least one does not, see Creating a default floating IP network and Creating a default provider network.

NOTE
If the Neutron trunk service plug-in is enabled, a trunk port is created by default. For more information, see Neutron trunk port.

9.4.11. Enabling access to the environment
At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

9.4.11.1. Enabling access with floating IP addresses
Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API, cluster applications, and the bootstrap process.

Procedure
1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>" <external_network>
```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>" <external_network>
```

3. By using the Red Hat OpenStack Platform (RHOSP) CLI, create the bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

```
api.<cluster_name>.<base_domain>. IN A <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
```
NOTE

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

5. Add the FIPs to the `inventory.yaml` file as the values of the following variables:

- `os_api_fip`
- `os_bootstrap_fip`
- `os_ingress_fip`

If you use these values, you must also enter an external network as the value of the `os_external_network` variable in the `inventory.yaml` file.

TIP

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

9.4.11.2. Completing installation without floating IP addresses

You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `inventory.yaml` file, do not define the following variables:

- `os_api_fip`
- `os_bootstrap_fip`
- `os_ingress_fip`

If you cannot provide an external network, you can also leave `os_external_network` blank. If you do not provide a value for `os_external_network`, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. Later in the installation process, when you create network resources, you must configure external connectivity on your own.

If you run the installer with the `wait-for` command from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.
NOTE
You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

\[
\text{api.<cluster}\text{.name>.<base}\text{.domain>. IN A <api}\text{.port}\text{.IP>}
\]

\[
*.apps.<cluster}\text{.name>.<base}\text{.domain>. IN A <ingress}\text{.port}\text{.IP>}
\]

If you do not control the DNS server, you can add the record to your /etc/hosts file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

9.4.12. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called clouds.yaml. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the clouds.yaml file:
   - If your RHOSP distribution includes the Horizon web UI, generate a clouds.yaml file in it.

     **IMPORTANT**
     Remember to add a password to the auth field. You can also keep secrets in a separate file from clouds.yaml.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about clouds.yaml, see Config files in the RHOSP documentation.

```
clouds:
  shiftstack:
    auth:
      project_name: shiftstack
      username: shiftstack_user
      password: XXX
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
    region_name: RegionOne
    auth:
      username: 'devuser'
      password: XXX
      project_name: 'devonly'
```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
   a. Copy the certificate authority file to your machine.
b. Add the machine to the certificate authority trust bundle:

```
$ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
```

c. Update the trust bundle:

```
$ sudo update-ca-trust extract
```

d. Add the `cacerts` key to the `clouds.yaml` file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

TIP

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the `ca-cert.pem` key in the `cloud-provider-config` keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the `clouds.yaml` file in one of the following locations:

a. The value of the `OS_CLIENT_CONFIG_FILE` environment variable

b. The current directory

c. A Unix-specific user configuration directory, for example `~/.config/openstack/clouds.yaml`

d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`

The installation program searches for `clouds.yaml` in that order.

### 9.4.13. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>
```
1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `openstack` as the platform to target.

iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

iv. Specify the floating IP address to use for external access to the OpenShift API.

v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane and compute nodes.

vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

You now have the file `install-config.yaml` in the directory that you specified.

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 9.4.14.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiVersion</code></td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td><code>baseDomain</code></td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td><code>metadata</code></td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
</tbody>
</table>
### Parameter: metadata.name

The name of the cluster. DNS records for the cluster are all subdomains of 

\{.metadata.name\}. \{.baseDomain\}.

String of lowercase letters, hyphens (-), and periods (.), such as `dev`. The string must be 14 characters or fewer long.

### Parameter: platform

The configuration for the specific platform upon which to perform the installation: 

`aws, baremetal, azure, openstack, ovirt, vsphere`. 

For additional information about platform.<platform> parameters, consult the following table for your specific platform.

### Parameter: pullSecret

Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 9.4.14.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 9.21. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <strong>10.128.0.0/14</strong> with a host prefix of <strong>/23</strong>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given <code>cidr</code>. A <code>hostPrefix</code> value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is <strong>172.30.0.0/16</strong>.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td></td>
</tr>
</tbody>
</table>

**Chapter 9. Installing on OpenStack**
### networking.machine Network
The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

<table>
<thead>
<tr>
<th>Networking.machin eNetwork</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An array of objects. For example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>networking:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>machineNetwork:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- cidr:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10.0.0.0/16</td>
<td></td>
</tr>
</tbody>
</table>

### networking.machine Network.cidr
Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.

<table>
<thead>
<tr>
<th>Networking.machineNetwork.cidr</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**
Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

### 9.4.14.3. Optional configuration parameters
Optional installation configuration parameters are described in the following table:

#### Table 9.22. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <strong>compute</strong>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <strong>compute</strong>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <strong>controlPlane.platform</strong> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <strong>MachinePool</strong> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
## controlPlane.hypertreading
Whether to enable or disable simultaneous multithreading, or **hypertreading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hypertreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use <strong>controlPlane</strong>. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use <strong>controlPlane</strong>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <strong>compute.platform</strong> parameter value.</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere</strong>, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>
## Parameter Description Values

### fips
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**
The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**
If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>imageContentSources</th>
<th>Sources and repositories for the release-image content.</th>
<th>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**Internal** or **External**. The default value is **External**.

Setting this field to **Internal** is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to **Internal**, the cluster will become non-functional. For more information, refer to BZ#1953035.

**sshKey**

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

One or more keys. For example:

```
sshKey:
<key1>
<key2>
<key3>
```

### 9.4.14.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:

**Table 9.23. Additional RHOSP parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.rootVolume.size</code></td>
<td>For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example <strong>30</strong>.</td>
</tr>
<tr>
<td><code>compute.platform.openstack.rootVolume.type</code></td>
<td>For compute machines, the root volume’s type.</td>
<td>String, for example <strong>performance</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>controlPlane.plat form.openstack.rootVolume.size</td>
<td>For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td>controlPlane.plat form.openstack.rootVolume.type</td>
<td>For control plane machines, the root volume’s type.</td>
<td>String, for example performance.</td>
</tr>
<tr>
<td>platform.openstack.cloud</td>
<td>The name of the RHOSP cloud to use from the list of clouds in the clouds.yaml file.</td>
<td>String, for example MyCloud.</td>
</tr>
<tr>
<td>platform.openstack.externalNetwork</td>
<td>The RHOSP external network name to be used for installation.</td>
<td>String, for example external.</td>
</tr>
<tr>
<td>platform.openstack.computeFlavor</td>
<td>The RHOSP flavor to use for control plane and compute machines.</td>
<td>String, for example m1.xlarge.</td>
</tr>
</tbody>
</table>

9.4.14.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

Table 9.24. Optional RHOSP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.platform.openstack.additionalNetworkIDs</td>
<td>Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td>compute.platform.openstack.additionalSecurityGroupIDs</td>
<td>Additional security groups that are associated with compute machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| `compute.platform.openstack.zones` | RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured.  
On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property. | A list of strings. For example, ["zone-1", "zone-2"]. |
| `controlPlane.platform.openstack.additionalNetworkIDs` | Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.                                                                                       | A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.                                                                                                           |
| `controlPlane.platform.openstack.additionalSecurityGroupIDs` | Additional security groups that are associated with control plane machines.                                                                                                                                   | A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.                                                                                                             |
| `controlPlane.platform.openstack.zones` | RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured.  
On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property. | A list of strings. For example, ["zone-1", "zone-2"]. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.openstack.clusterOSImage</code></td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with an SHA-256 checksum.                                                                                                                                     For example, <a href="http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d">http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</a>. The value can also be the name of an existing Glance image, for example my-rhcos.</td>
</tr>
<tr>
<td><code>platform.openstack.defaultMachinePlatform</code></td>
<td>The default machine pool platform configuration.</td>
<td>{                                                                                           &quot;type&quot;: &quot;ml.large&quot;, &quot;rootVolume&quot;: { &quot;size&quot;: 30, &quot;type&quot;: &quot;performance&quot; } }</td>
</tr>
<tr>
<td><code>platform.openstack.ingressFloatingIP</code></td>
<td>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the <code>platform.openstack.externalNetwork</code> property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><code>platform.openstack.lbFloatingIP</code></td>
<td>An existing floating IP address to associate with the API load balancer. To use this property, you must also define the <code>platform.openstack.externalNetwork</code> property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><code>platform.openstack.externalDNS</code></td>
<td>IP addresses for external DNS servers that cluster instances use for DNS resolution.</td>
<td>A list of IP addresses as strings. For example, [&quot;8.8.8.8&quot;, &quot;192.168.1.12&quot;].</td>
</tr>
</tbody>
</table>
The UUID of a RHOSP subnet that the cluster’s nodes use. Nodes and virtual IP (VIP) ports are created on this subnet.

The first item in `networking.machinemachineNetwork` must match the value of `machinesSubnet`.

If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.

A UUID as a string. For example, `fa806b2f-ac49-4bce-b9db-124bc64209bf`.

### 9.4.14.6. Custom subnets in RHOSP deployments

Optionally, you can deploy a cluster on a Red Hat OpenStack Platform (RHOSP) subnet of your choice. The subnet’s GUID is passed as the value of `platform.openstack.machinesSubnet` in the `install-config.yaml` file.

This subnet is used as the cluster’s primary subnet; nodes and ports are created on it.

Before you run the OpenShift Container Platform installer with a custom subnet, verify that:

- The target network and subnet are available.
- DHCP is enabled on the target subnet.
- You can provide installer credentials that have permission to create ports on the target network.
- If your network configuration requires a router, it is created in RHOSP. Some configurations rely on routers for floating IP address translation.
- Your network configuration does not rely on a provider network. Provider networks are not supported.

**NOTE**

By default, the API VIP takes x.x.x.5 and the Ingress VIP takes x.x.x.7 from your network’s CIDR block. To override these default values, set values for `platform.openstack.apiVIP` and `platform.openstack.ingressVIP` that are outside of the DHCP allocation pool.

### 9.4.14.7. Sample customized `install-config.yaml` file for RHOSP with Kuryr

To deploy with Kuryr SDN instead of the default OpenShift SDN, you must modify the `install-config.yaml` file to include `Kuryr` as the desired `networking.networkType` and proceed with the default OpenShift Container Platform SDN installation steps. This sample `install-config.yaml` demonstrates all
of the possible Red Hat OpenStack Platform (RHOSP) customization options.

**IMPORTANT**

This sample file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program.

```yaml
apiVersion: v1
baseDomain: example.com
clusterID: os-test
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
  - name: worker
    platform:
      openstack:
        type: ml.large
        replicas: 3
    metadata:
      name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: Kuryr
  platform:
    openstack:
      cloud: mycloud
      externalNetwork: external
      computeFlavor: m1.xlarge
      lbFloatingIP: 128.0.0.1
      trunkSupport: true
      octaviaSupport: true
      pullSecret: '{"auths": ...}'
      sshKey: ssh-ed25519 AAAA...
```

1. The Amphora Octavia driver creates two ports per load balancer. As a result, the service subnet that the installer creates is twice the size of the CIDR that is specified as the value of the `serviceNetwork` property. The larger range is required to prevent IP address conflicts.

2.3. Both `trunkSupport` and `octaviaSupport` are automatically discovered by the installer, so there is no need to set them. But if your environment does not meet both requirements, Kuryr SDN will not properly work. Trunks are needed to connect the pods to the RHOSP network and Octavia is required to create the OpenShift Container Platform services.

A Kuryr ports pool maintains a number of ports on standby for pod creation. Keeping ports on standby minimizes pod creation time. Without ports pools, Kuryr must explicitly request port creation or deletion whenever a pod is created or deleted.

The Neutron ports that Kuryr uses are created in subnets that are tied to namespaces. These pod ports are also added as subports to the primary port of OpenShift Container Platform cluster nodes.

Because Kuryr keeps each namespace in a separate subnet, a separate ports pool is maintained for each namespace-worker pair.

Prior to installing a cluster, you can set the following parameters in the `cluster-network-03-config.yml` manifest file to configure ports pool behavior:

- The `enablePortPoolsPrepopulation` parameter controls pool prepopulation, which forces Kuryr to add ports to the pool when it is created, such as when a new host is added, or a new namespace is created. The default value is `false`.
- The `poolMinPorts` parameter is the minimum number of free ports that are kept in the pool. The default value is `1`.
- The `poolMaxPorts` parameter is the maximum number of free ports that are kept in the pool. A value of `0` disables that upper bound. This is the default setting. If your OpenStack port quota is low, or you have a limited number of IP addresses on the pod network, consider setting this option to ensure that unneeded ports are deleted.
- The `poolBatchPorts` parameter defines the maximum number of Neutron ports that can be created at once. The default value is `3`.

9.4.14.9. Adjusting Kuryr ports pools during installation

During installation, you can configure how Kuryr manages Red Hat OpenStack Platform (RHOSP) Neutron ports to control the speed and efficiency of pod creation.

**Prerequisites**

- Create and modify the `install-config.yaml` file.

**Procedure**

1. From a command line, create the manifest files:

   ```sh
   $ ./openshift-install create manifests --dir <installation_directory>
   ``

   For `<installation_directory>`, specify the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a file that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```sh
   $ touch <installation_directory>/manifests/cluster-network-03-config.yml
   ``

   For `<installation_directory>`, specify the directory name that contains the `manifests/` directory for your cluster.
After creating the file, several network configuration files are in the `manifests/` directory, as shown:

```bash
$ ls <installation_directory>/manifests/cluster-network-*/
```

Example output

```
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. Open the `cluster-network-03-config.yml` file in an editor, and enter a custom resource (CR) that describes the Cluster Network Operator configuration that you want:

```bash
$ oc edit networks.operator.openshift.io cluster
```

4. Edit the settings to meet your requirements. The following file is provided as an example:

```yaml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  serviceNetwork:
    - 172.30.0.0/16
  defaultNetwork:
    type: Kuryr
    kuryrConfig:
      enablePortPoolsPrepopulation: false
      poolMinPorts: 1
      poolBatchPorts: 3
      poolMaxPorts: 5
      openstackServiceNetwork: 172.30.0.0/15
```

1. Set the value of `enablePortPoolsPrepopulation` to `true` to make Kuryr create new Neutron ports after a namespace is created or a new node is added to the cluster. This setting raises the Neutron ports quota but can reduce the time that is required to spawn pods. The default value is `false`.

2. Kuryr creates new ports for a pool if the number of free ports in that pool is lower than the value of `poolMinPorts`. The default value is `1`.

3. `poolBatchPorts` controls the number of new ports that are created if the number of free ports is lower than the value of `poolMinPorts`. The default value is `3`.

4. If the number of free ports in a pool is higher than the value of `poolMaxPorts`, Kuryr deletes them until the number matches that value. Setting this value to `0` disables this upper bound, preventing pools from shrinking. The default value is `0`.

5. The `openStackServiceNetwork` parameter defines the CIDR range of the network from which IP addresses are allocated to RHOSP Octavia’s LoadBalancers.
If this parameter is used with the Amphora driver, Octavia takes two IP addresses from this network for each load balancer: one for OpenShift and the other for VRRP connections. Because these IP addresses are managed by OpenShift Container Platform and Neutron respectively, they must come from different pools. Therefore, the value of `openStackServiceNetwork` must be at least twice the size of the value of `serviceNetwork`, and the value of `serviceNetwork` must overlap entirely with the range that is defined by `openStackServiceNetwork`.

The CNO verifies that VRRP IP addresses that are taken from the range that is defined by this parameter do not overlap with the range that is defined by the `serviceNetwork` parameter.

If this parameter is not set, the CNO uses an expanded value of `serviceNetwork` that is determined by decrementing the prefix size by 1.

5. Save the `cluster-network-03-config.yml` file, and exit the text editor.

6. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory while creating the cluster.

9.4.14.10. Setting a custom subnet for machines

The IP range that the installation program uses by default might not match the Neutron subnet that you create when you install OpenShift Container Platform. If necessary, update the CIDR value for new machines by editing the installation configuration file.

**Prerequisites**

- You have the `install-config.yaml` file that was generated by the OpenShift Container Platform installation program.

**Procedure**

1. On a command line, browse to the directory that contains `install-config.yaml`.

2. From that directory, either run a script to edit the `install-config.yaml` file or update the file manually:
   - To set the value by using a script, run:
     ```bash
     $ python -c '
     import yaml;
     path = "install-config.yaml";
     data = yaml.safe_load(open(path));
     data['networking']['machineNetwork'] = [{"cidr": "192.168.0.0/18"}];
     open(path, "w").write(yaml.dump(data, default_flow_style=False))'
     ```
     • Insert a value that matches your intended Neutron subnet, e.g. `192.0.2.0/24`.
   - To set the value manually, open the file and set the value of `networking.machineCIDR` to something that matches your intended Neutron subnet.

9.4.14.11. Emptying compute machine pools
To proceed with an installation that uses your own infrastructure, set the number of compute machines in the installation configuration file to zero. Later, you create these machines manually.

**Prerequisites**

- You have the *install-config.yaml* file that was generated by the OpenShift Container Platform installation program.

**Procedure**

1. On a command line, browse to the directory that contains *install-config.yaml*.
2. From that directory, either run a script to edit the *install-config.yaml* file or update the file manually:
   - To set the value by using a script, run:
     ```
     $ python -c 'import yaml; 
     path = "install-config.yaml";
     data = yaml.safe_load(open(path));
     data["compute"][0]["replicas"] = 0;
     open(path, "w").write(yaml.dump(data, default_flow_style=False))'
     ```
   - To set the value manually, open the file and set the value of `compute.<first entry>.replicas` to 0.

9.4.14.12. Modifying the network type

By default, the installation program selects the OpenShiftSDN network type. To use Kuryr instead, change the value in the installation configuration file that the program generated.

**Prerequisites**

- You have the file *install-config.yaml* that was generated by the OpenShift Container Platform installation program.

**Procedure**

1. In a command prompt, browse to the directory that contains *install-config.yaml*.
2. From that directory, either run a script to edit the *install-config.yaml* file or update the file manually:
   - To set the value by using a script, run:
     ```
     $ python -c 'import yaml; 
     path = "install-config.yaml";
     data = yaml.safe_load(open(path));
     data["networking"]["networkType"] = "Kuryr";
     open(path, "w").write(yaml.dump(data, default_flow_style=False))'
     ```
   - To set the value manually, open the file and set `networking.networkType` to "Kuryr".
9.4.15. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.
- You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   **Example output**

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   **For `<installation_directory>`**, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```bash
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*yaml openshift/99_openshift-cluster-api_worker-machineset-*yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.
You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   - `auth`
     - `kubeadmin-password`
     - `kubeconfig`
   - `bootstrap.ign`
   - `master.ign`
   - `metadata.json`
   - `worker.ign`

5. Export the metadata file’s `infraID` key as an environment variable:

   ```bash
   $ export INFRA_ID=$(jq -r .infraID metadata.json)
   ```

**TIP**

Extract the `infraID` key from `metadata.json` and use it as a prefix for all of the RHOSP resources that you create. By doing so, you avoid name conflicts when making multiple deployments in the same project.

### 9.4.16. Preparing the bootstrap Ignition files

The OpenShift Container Platform installation process relies on bootstrap machines that are created from a bootstrap Ignition configuration file.

Edit the file and upload it. Then, create a secondary bootstrap Ignition configuration file that Red Hat OpenStack Platform (RHOSP) uses to download the primary file.

**Prerequisites**

- You have the bootstrap Ignition file that the installer program generates, `bootstrap.ign`. 
The infrastructure ID from the installer’s metadata file is set as an environment variable ($INFRA_ID).

- If the variable is not set, see Creating the Kubernetes manifest and Ignition config files.

- You have an HTTP(S)-accessible way to store the bootstrap Ignition file.
  - The documented procedure uses the RHOSP image service (Glance), but you can also use the RHOSP storage service (Swift), Amazon S3, an internal HTTP server, or an ad hoc Nova server.

**Procedure**

1. Run the following Python script. The script modifies the bootstrap Ignition file to set the hostname and, if available, CA certificate file when it runs:

```python
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-bootstrap
').decode().strip()
files.append(
    {'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }}
)

cert_path = os.environ.get('OS_CACERT', '')
if cert_path:
    with open(cert_path, 'r') as f:
        cert = f.read().encode()
        cert_b64 = base64.standard_b64encode(cert).decode().strip()

    files.append(
        {'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
            'source': 'data:text/plain;charset=utf-8;base64,' + cert_b64
        }}
    )

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```
2. Using the RHOSP CLI, create an image that uses the bootstrap Ignition file:

   ```
   $ openstack image create --disk-format=raw --container-format=bare --file bootstrap.ign <image_name>
   ```

3. Get the image’s details:

   ```
   $ openstack image show <image_name>
   ```

   Make a note of the `file` value; it follows the pattern `v2/images/<image_ID>/file`.

   **NOTE**
   Verify that the image you created is active.

4. Retrieve the image service’s public address:

   ```
   $ openstack catalog show image
   ```

5. Combine the public address with the image `file` value and save the result as the storage location. The location follows the pattern


6. Generate an auth token and save the token ID:

   ```
   $ openstack token issue -c id -f value
   ```

7. Insert the following content into a file called `$INFRA_ID-bootstrap-ignition.json` and edit the placeholders to match your own values:

   ```json
   {
   "ignition": {
   "config": {
   "merge": [{
   "source": "<storage_url>",
   "httpHeaders": [{
   "name": "X-Auth-Token",
   "value": "<token_ID>"
   }]
   }
   },
   "security": {
   "tls": {
   "certificateAuthorities": [{
   "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>"
   }]
   },
   "version": "3.1.0"
   }
   }
   ```
1. Replace the value of `ignition.config.merge.source` with the bootstrap Ignition file storage URL.

2. Set `name` in `httpHeaders` to "X-Auth-Token".

3. Set `value` in `httpHeaders` to your token’s ID.

4. If the bootstrap Ignition file server uses a self-signed certificate, include the base64-encoded certificate.

8. Save the secondary Ignition config file.

The bootstrap Ignition data will be passed to RHOSP during installation.

---

8. The bootstrap Ignition file contains sensitive information, like `clouds.yaml` credentials. Ensure that you store it in a secure place, and delete it after you complete the installation process.

---

### 9.4.17. Creating control plane Ignition config files on RHOSP

Installing OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) on your own infrastructure requires control plane Ignition config files. You must create multiple config files.

#### NOTE

As with the bootstrap Ignition configuration, you must explicitly define a hostname for each control plane machine.

#### Prerequisites

- The infrastructure ID from the installation program’s metadata file is set as an environment variable (`$INFRA_ID`).
  - If the variable is not set, see "Creating the Kubernetes manifest and Ignition config files."

#### Procedure

- On a command line, run the following Python script:

```bash
$ for index in $(seq 0 2); do
  MASTER_HOSTNAME="$INFRA_ID-master-$index"
  python -c "import base64, json, sys; ignition = json.load(sys.stdin); storage = ignition.get('storage', {}); files = storage.get('files', []); files.append({"path": '/etc/hostname', 'mode': 420, 'contents': {'source': 'data:text/plain;charset=utf-8;base64,' + base64.standard_b64encode(b'${MASTER_HOSTNAME}').decode().strip(), 'verification': {}},
```
You now have three control plane Ignition files: `<INFRA_ID>-master-0-ignition.json`, `<INFRA_ID>-master-1-ignition.json`, and `<INFRA_ID>-master-2-ignition.json`.

### 9.4.18. Creating network resources on RHOSP

Create the network resources that an OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) installation on your own infrastructure requires. To save time, run supplied Ansible playbooks that generate security groups, networks, subnets, routers, and ports.

#### Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."

#### Procedure

1. Optional: Add an external network value to the `inventory.yaml` playbook:

   **Example external network value in the inventory.yaml Ansible playbook**

   ```yaml
   ... # The public network providing connectivity to the cluster. If not provided, the cluster external connectivity must be provided in another way.
   # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
   os_external_network: 'external'
   ...
   ```

   **IMPORTANT**

   If you did not provide a value for `os_external_network` in the `inventory.yaml` file, you must ensure that VMs can access Glance and an external connection yourself.

2. Optional: Add external network and floating IP (FIP) address values to the `inventory.yaml` playbook:

   **Example FIP values in the inventory.yaml Ansible playbook**

   ```yaml
   ... # OpenShift API floating IP address. If this value is non-empty, the corresponding floating IP will be attached to the Control Plane to serve the OpenShift API.
   os_api_fip: '203.0.113.23'
   ```
IMPORTANT

If you do not define values for `os_api_fip` and `os_ingress_fip`, you must perform post-installation network configuration.

If you do not define a value for `os_bootstrap_fip`, the installer cannot download debugging information from failed installations.

See "Enabling access to the environment" for more information.

3. On a command line, create security groups by running the `security-groups.yaml` playbook:

   $ ansible-playbook -i inventory.yaml security-groups.yaml

4. On a command line, create a network, subnet, and router by running the `network.yaml` playbook:

   $ ansible-playbook -i inventory.yaml network.yaml

5. Optional: If you want to control the default resolvers that Nova servers use, run the RHOSP CLI command:

   $ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2> "$INFRA_ID-nodes"

9.4.19. Creating the bootstrap machine on RHOSP

Create a bootstrap machine and give it the network access it needs to run on Red Hat OpenStack Platform (RHOSP). Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The `inventory.yaml`, `common.yaml`, and `bootstrap.yaml` Ansible playbooks are in a common directory.
- The `metadata.json` file that the installation program created is in the same directory as the Ansible playbooks.
Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. On a command line, run the `bootstrap.yaml` playbook:

   ```bash
   $ ansible-playbook -i inventory.yaml bootstrap.yaml
   ```

3. After the bootstrap server is active, view the logs to verify that the Ignition files were received:

   ```bash
   $ openstack console log show "$INFRA_ID-bootstrap"
   ```

9.4.20. Creating the control plane machines on RHOSP

Create three control plane machines by using the Ignition config files that you generated. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The infrastructure ID from the installation program’s metadata file is set as an environment variable (`$INFRA_ID`).
- The `inventory.yaml`, `common.yaml`, and `control-plane.yaml` Ansible playbooks are in a common directory.
- You have the three Ignition files that were created in "Creating control plane Ignition config files."

Procedure

1. On a command line, change the working directory to the location of the playbooks.

2. If the control plane Ignition config files aren’t already in your working directory, copy them into it.

3. On a command line, run the `control-plane.yaml` playbook:

   ```bash
   $ ansible-playbook -i inventory.yaml control-plane.yaml
   ```

4. Run the following command to monitor the bootstrapping process:

   ```bash
   $ openshift-install wait-for bootstrap-complete
   ```

   You will see messages that confirm that the control plane machines are running and have joined the cluster:

   ```bash
   INFO API v1.14.6+f9b5405 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   ... INFO It is now safe to remove the bootstrap resources
   ```
9.4.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**
   
   `system:admin`

9.4.22. Deleting bootstrap resources from RHOSP

Delete the bootstrap resources that you no longer need.

**Prerequisites**

- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The `inventory.yaml`, `common.yaml`, and `down-bootstrap.yaml` Ansible playbooks are in a common directory.
- The control plane machines are running.
  - If you do not know the status of the machines, see "Verifying cluster status."

**Procedure**

1. On a command line, change the working directory to the location of the playbooks.

2. On a command line, run the `down-bootstrap.yaml` playbook:
The bootstrap port, server, and floating IP address are deleted.

WARNING
If you did not disable the bootstrap Ignition file URL earlier, do so now.

9.4.23. Creating compute machines on RHOSP

After standing up the control plane, create compute machines. Red Hat provides an Ansible playbook that you run to simplify this process.

Prerequisites
- You downloaded the modules in "Downloading playbook dependencies."
- You downloaded the playbooks in "Downloading the installation playbooks."
- The `inventory.yaml`, `common.yaml`, and `compute-nodes.yaml` Ansible playbooks are in a common directory.
- The `metadata.json` file that the installation program created is in the same directory as the Ansible playbooks.
- The control plane is active.

Procedure
1. On a command line, change the working directory to the location of the playbooks.
2. On a command line, run the playbook:

   `$ ansible-playbook -i inventory.yaml down-bootstrap.yaml`

   `$ ansible-playbook -i inventory.yaml compute-nodes.yaml`

Next steps
- Approve the certificate signing requests for the machines.

9.4.24. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites
- You added machines to your cluster.
Procedure

1. Confirm that the cluster recognizes the machines:

   $ oc get nodes

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   $ oc get csr

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

   **NOTE**

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.
NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the oc exec, oc rsh, and oc logs commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the system:node or system:admin groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  $ oc adm certificate approve <csr_name> ①

  ① <csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

  $ oc get csr

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  $ oc adm certificate approve <csr_name> ①

  ① <csr_name> is the name of a CSR from the list of current CSRs.
- To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```bash
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

### 9.4.25. Verifying a successful installation

Verify that the OpenShift Container Platform installation is complete.

**Prerequisites**

- You have the installation program (**openshift-install**)

**Procedure**

- On a command line, enter:

  ```bash
  $ openshift-install --log-level debug wait-for install-complete
  ```

The program outputs the console URL, as well as the administrator’s login information.

### 9.4.26. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to **OpenShift Cluster Manager**.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

9.4.27. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- If you need to enable external access to node ports, configure ingress cluster traffic by using a node port.
- If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses.

9.5. INSTALLING A CLUSTER ON OPENSTACK IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.6, you can install a cluster on Red Hat OpenStack Platform (RHOSP) in a restricted network by creating an internal mirror of the installation release content.

NOTE

Installing in a restricted network is supported only for installer-provisioned installations.

Prerequisites

- Create a registry on your mirror host and obtain the imageContentSources data for your version of OpenShift Container Platform.

IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- Review details about the OpenShift Container Platform installation and update processes.
  - Verify that OpenShift Container Platform 4.6 is compatible with your RHOSP version by consulting the architecture documentation’s list of available platforms. You can also compare platform support across different versions by viewing the OpenShift Container Platform on RHOSP support matrix.
  - Verify that your network configuration does not rely on a provider network. Provider networks are not supported.
  - Have the metadata service enabled in RHOSP.

9.5.1. About installations in restricted networks
In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

9.5.1.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The ClusterVersion status includes an Unable to retrieve available updates error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

9.5.2. Resource guidelines for installing OpenShift Container Platform on RHOSP

To support an OpenShift Container Platform installation, your Red Hat OpenStack Platform (RHOSP) quota must meet the following requirements:

Table 9.25. Recommended resources for a default OpenShift Container Platform cluster on RHOSP

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating IP addresses</td>
<td>3</td>
</tr>
<tr>
<td>Ports</td>
<td>15</td>
</tr>
<tr>
<td>Routers</td>
<td>1</td>
</tr>
<tr>
<td>Subnets</td>
<td>1</td>
</tr>
<tr>
<td>RAM</td>
<td>112 GB</td>
</tr>
<tr>
<td>vCPUs</td>
<td>28</td>
</tr>
<tr>
<td>Volume storage</td>
<td>275 GB</td>
</tr>
<tr>
<td>Instances</td>
<td>7</td>
</tr>
<tr>
<td>Security groups</td>
<td>3</td>
</tr>
<tr>
<td>Security group rules</td>
<td>60</td>
</tr>
</tbody>
</table>
A cluster might function with fewer than recommended resources, but its performance is not guaranteed.

**IMPORTANT**

If RHOSP object storage (Swift) is available and operated by a user account with the `swiftoperator` role, it is used as the default backend for the OpenShift Container Platform image registry. In this case, the volume storage requirement is 175 GB. Swift space requirements vary depending on the size of the image registry.

**NOTE**

By default, your security group and security group rule quotas might be low. If you encounter problems, run `openstack quota set --secgroups 3 --secgroup-rules 60 <project>` as an administrator to increase them.

An OpenShift Container Platform deployment comprises control plane machines, compute machines, and a bootstrap machine.

### 9.5.2.1. Control plane and compute machines

By default, the OpenShift Container Platform installation process stands up three control plane and three compute machines.

Each machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space

**TIP**

Compute machines host the applications that you run on OpenShift Container Platform; aim to run as many as you can.

### 9.5.2.2. Bootstrap machine

During installation, a bootstrap machine is temporarily provisioned to stand up the control plane. After the production control plane is ready, the bootstrap machine is deprovisioned.

The bootstrap machine requires:

- An instance from the RHOSP quota
- A port from the RHOSP quota
- A flavor with at least 16 GB memory, 4 vCPUs, and 25 GB storage space
9.5.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

9.5.4. Enabling Swift on RHOSP

Swift is operated by a user account with the **swiftoperator** role. Add the role to an account before you run the installation program.

**IMPORTANT**

If the Red Hat OpenStack Platform (RHOSP) object storage service, commonly known as Swift, is available, OpenShift Container Platform uses it as the image registry storage. If it is unavailable, the installation program relies on the RHOSP block storage service, commonly known as Cinder.

If Swift is present and you want to use it, you must enable access to it. If it is not present, or if you do not want to use it, skip this section.

**Prerequisites**

- You have a RHOSP administrator account on the target environment.

- The Swift service is installed.

- On Ceph RGW, the **account in url** option is enabled.

**Procedure**

To enable Swift on RHOSP:

1. As an administrator in the RHOSP CLI, add the **swiftoperator** role to the account that will access Swift:

   ```bash
   $ openstack role add --user <user> --project <project> swiftoperator
   ```
Your RHOSP deployment can now use Swift for the image registry.

9.5.5. Defining parameters for the installation program

The OpenShift Container Platform installation program relies on a file that is called `clouds.yaml`. The file describes Red Hat OpenStack Platform (RHOSP) configuration parameters, including the project name, log in information, and authorization service URLs.

Procedure

1. Create the `clouds.yaml` file:
   - If your RHOSP distribution includes the Horizon web UI, generate a `clouds.yaml` file in it.

   **IMPORTANCE**

   Remember to add a password to the `auth` field. You can also keep secrets in a separate file from `clouds.yaml`.

   - If your RHOSP distribution does not include the Horizon web UI, or you do not want to use Horizon, create the file yourself. For detailed information about `clouds.yaml`, see Config files in the RHOSP documentation.

     ```yaml
     clouds:
     shiftstack:
     auth:
       project_name: shiftstack
       username: shiftstack_user
       password: XXX
       user_domain_name: Default
       project_domain_name: Default
     dev-env:
       region_name: RegionOne
       auth:
         username: 'devuser'
         password: XXX
         project_name: 'devonly'
     ```

2. If your RHOSP installation uses self-signed certificate authority (CA) certificates for endpoint authentication:
   a. Copy the certificate authority file to your machine.
   b. Add the machine to the certificate authority trust bundle:

      ```
      $ sudo cp ca.crt.pem /etc/pki/ca-trust/source/anchors/
      ```
   c. Update the trust bundle:

      ```
      $ sudo update-ca-trust extract
      ```
d. Add the `cacerts` key to the `clouds.yaml` file. The value must be an absolute, non-root-accessible path to the CA certificate:

```
clouds:
  shiftstack:
    ...
  cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

**TIP**

After you run the installer with a custom CA certificate, you can update the certificate by editing the value of the `ca-cert.pem` key in the `cloud-provider-config` keymap. On a command line, run:

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. Place the `clouds.yaml` file in one of the following locations:
   
   a. The value of the `OS_CLIENT_CONFIG_FILE` environment variable
   
   b. The current directory
   
   c. A Unix-specific user configuration directory, for example `~/.config/openstack/clouds.yaml`
   
   d. A Unix-specific site configuration directory, for example `/etc/openstack/clouds.yaml`
   
   The installation program searches for `clouds.yaml` in that order.

### 9.5.6. Creating the RHCOS image for restricted network installations

Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network Red Hat OpenStack Platform (RHOSP) environment.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

**Procedure**


2. Under **Version**, select the most recent release of OpenShift Container Platform 4.6 for RHEL 8.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) – OpenStack Image (QCOW)** image.
4. Decompress the image.

**NOTE**

You must decompress the image before the cluster can use it. The name of the downloaded file might not contain a compression extension, like `.gz` or `.tgz`. To find out if or how the file is compressed, in a command line, enter:

```
$ file <name_of_downloaded_file>
```

5. Upload the image that you decompressed to a location that is accessible from the bastion server, like Glance. For example:

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --disk-format qcow2 rhcos-${RHCOS_VERSION}
```

**IMPORTANT**

Depending on your RHOSP environment, you might be able to upload the image in either `.raw` or `.qcow2` formats. If you use Ceph, you must use the `.raw` format.

**WARNING**

If the installation program finds multiple images with the same name, it chooses one of them at random. To avoid this behavior, create unique names for resources in RHOSP.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

**9.5.7. Creating the installation configuration file**

You can customize the OpenShift Container Platform cluster you install on Red Hat OpenStack Platform (RHOSP).

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- Have the `imageContentSources` values that were generated during mirror registry creation.
- Obtain the contents of the certificate for your mirror registry.
- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.

**Procedure**
1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   **IMPORTANT**

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

      **NOTE**

      For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

      ii. Select `openstack` as the platform to target.

      iii. Specify the Red Hat OpenStack Platform (RHOSP) external network name to use for installing the cluster.

      iv. Specify the floating IP address to use for external access to the OpenShift API.

      v. Specify a RHOSP flavor with at least 16 GB RAM to use for control plane and compute nodes.

      vi. Select the base domain to deploy the cluster to. All DNS records will be sub-domains of this base and will also include the cluster name.

      vii. Enter a name for your cluster. The name must be 14 or fewer characters long.

      viii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. In the `install-config.yaml` file, set the value of `platform.openstack.clusterOSImage` to the image location or name. For example:

   ```yaml
   platform:
     openstack:
       clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
   ```
3. Edit the `install-config.yaml` file to provide the additional information that is required for an installation in a restricted network.
   
   a. Update the **pullSecret** value to contain the authentication information for your registry:
      
      ```yaml
      pullSecret: {
        "auths": {
          "<mirror_host_name>:5000": {
            "auth": "<credentials>",
            "email": "you@example.com"
          }
        }
      }
      ```
      
      For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.
   
   b. Add the **additionalTrustBundle** parameter and value.
      
      ```yaml
      additionalTrustBundle: |
      -----BEGIN CERTIFICATE-----
      ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
      -----END CERTIFICATE-----
      ```
      
      The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.
   
   c. Add the image content resources, which look like this excerpt:
      
      ```yaml
      imageContentSources:
      - mirrors:
        - <mirror_host_name>:5000/<repo_name>/release
          source: quay.example.com/openshift-release-dev/ocp-release
        - mirrors:
          source: registry.example.com/ocp/release
      ```
      
      To complete these values, use the **imageContentSources** that you recorded during mirror registry creation.
   
4. Make any other modifications to the `install-config.yaml` file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.
   
5. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
   
   **IMPORTANT**
   
   The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

9.5.7.1. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations.
   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents
specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

**NOTE**

Only the Proxy object named cluster is supported, and no additional proxies can be created.

### 9.5.7.2. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the install-config.yaml file.

**IMPORTANT**

The openshift-install command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 9.5.7.2.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 9.26. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;, &lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}, {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev. The string must be 14 characters or fewer long.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td>{ &quot;auths&quot;:{  &quot;cloud.openshift.com&quot;:{    &quot;auth&quot;:&quot;b3Blb=&quot;,    &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;  },  &quot;quay.io&quot;:{    &quot;auth&quot;:&quot;b3Blb=&quot;,    &quot;email&quot;:&quot;<a href="mailto:you@example.com">you@example.com</a>&quot;  }  }  }</td>
</tr>
</tbody>
</table>

### CHAPTER 9. INSTALLING ON OPENSTACK

1551
9.5.7.2.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 9.27. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>You cannot modify parameters specified by the</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>networking</code> object after installation.</td>
<td></td>
</tr>
<tr>
<td>networking.networkType</td>
<td>The cluster network provider Container Network</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td></td>
<td>Interface (CNI) plug-in to install.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>10.128.0.0/14</code> with a host prefix of <code>/23</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The prefix length for an IPv4 block is between 0 and 32.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given cidr. A <code>hostPrefix</code> value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix.</td>
</tr>
<tr>
<td></td>
<td>The default value is 23.</td>
<td></td>
</tr>
</tbody>
</table>
### networking.serviceNetwork

The IP address block for services. The default value is `172.30.0.0/16`.

The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
| networking.serviceNetwork | The IP address block for services. The default value is **172.30.0.0/16**. | An array with an IP address block in CIDR format. For example:  
  ```yaml
  networking:
    serviceNetwork:
      - 172.30.0.0/16
  ``` |

### networking.machineNetwork

The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
| networking.machineNetwork | The IP address blocks for machines.                                           | An array of objects. For example:  
  ```yaml
  networking:
    machineNetwork:
      - cidr: **10.0.0.0/16**
  ``` |

### networking.machineNetwork.cidr

Required if you use `networking.machineNetwork`. An IP address block. The default value is **10.0.0.0/16** for all platforms other than libvirt. For libvirt, the default value is **192.168.126.0/24**.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
</table>
| networking.machineNetwork.cidr | Required if you use networking.machineNetwork. An IP address block. The default value is **10.0.0.0/16** for all platforms other than libvirt. For libvirt, the default value is **192.168.126.0/24**. | An IP network block in CIDR notation. For example, **10.0.0.0/16**.  
  ```yaml
  networking:
    machineNetwork:
      - cidr: 10.0.0.0/16
  ``` |

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

### 9.5.7.2.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or {}</code></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to <code>2</code>. The default value is <code>3</code>.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use <code>controlPlane</code>. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td>false or true</td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a source and, optionally, mirrors, as described in the following rows of this table.</td>
</tr>
</tbody>
</table>
Table 9.29. Additional RHOSP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example: sshKey: &lt;key1&gt; &lt;key2&gt; &lt;key3&gt;</td>
</tr>
</tbody>
</table>

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

9.5.7.2.4. Additional Red Hat OpenStack Platform (RHOSP) configuration parameters

Additional RHOSP configuration parameters are described in the following table:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.rootVolume.size</code></td>
<td>For compute machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td><code>compute.platform.openstack.rootVolume.type</code></td>
<td>For compute machines, the root volume’s type.</td>
<td>String, for example <code>performance</code>.</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.rootVolume.size</code></td>
<td>For control plane machines, the size in gigabytes of the root volume. If you do not set this value, machines use ephemeral storage.</td>
<td>Integer, for example 30.</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.rootVolume.type</code></td>
<td>For control plane machines, the root volume’s type.</td>
<td>String, for example <code>performance</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.cloud</code></td>
<td>The name of the RHOSP cloud to use from the list of clouds in the <code>clouds.yaml</code> file.</td>
<td>String, for example <code>MyCloud</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.externalNetwork</code></td>
<td>The RHOSP external network name to be used for installation.</td>
<td>String, for example <code>external</code>.</td>
</tr>
<tr>
<td><code>platform.openstack.computeFlavor</code></td>
<td>The RHOSP flavor to use for control plane and compute machines.</td>
<td>String, for example <code>m1.xlarge</code>.</td>
</tr>
</tbody>
</table>

### 9.5.7.2.5. Optional RHOSP configuration parameters

Optional RHOSP configuration parameters are described in the following table:

#### Table 9.30. Optional RHOSP parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.platform.openstack.additionalNetworkIDs</code></td>
<td>Additional networks that are associated with compute machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, <code>fa806b2f-ac49-4bce-b9db-124bc64209bf</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>compute.platform.openstack.additionalSecurityGroupIDs</code></td>
<td>Additional security groups that are associated with compute machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td><code>compute.platform.openstack.zones</code></td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.additionalNetworkIDs</code></td>
<td>Additional networks that are associated with control plane machines. Allowed address pairs are not created for additional networks.</td>
<td>A list of one or more UUIDs as strings. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
<tr>
<td><code>controlPlane.platform.openstack.additionalSecurityGroupIDs</code></td>
<td>Additional security groups that are associated with control plane machines.</td>
<td>A list of one or more UUIDs as strings. For example, 7ee219f3-d2e9-48a1-96c2-e7429f1b0da7.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane.platform.openstack.zones</td>
<td>RHOSP Compute (Nova) availability zones (AZs) to install machines on. If this parameter is not set, the installer relies on the default settings for Nova that the RHOSP administrator configured. On clusters that use Kuryr, RHOSP Octavia does not support availability zones. Load balancers and, if you are using the Amphora provider driver, OpenShift Container Platform services that rely on Amphora VMs, are not created according to the value of this property.</td>
<td>A list of strings. For example, [&quot;zone-1&quot;, &quot;zone-2&quot;].</td>
</tr>
<tr>
<td>platform.openstack.clusterOSImage</td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with an SHA-256 checksum. For example, <a href="http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d">http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d</a>. The value can also be the name of an existing Glance image, for example my-rhcos.</td>
</tr>
<tr>
<td>platform.openstack.defaultMachinePlatform</td>
<td>The default machine pool platform configuration.</td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;type&quot;: &quot;ml.large&quot;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;rootVolume&quot;: {</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;size&quot;: 30,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;type&quot;: &quot;performance&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td>platform.openstack.ingressFloatingIP</td>
<td>An existing floating IP address to associate with the Ingress port. To use this property, you must also define the platform.openstack.externalNetwork property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>platform.openstack.lbFloatingIP</td>
<td>An existing floating IP address to associate with the API load balancer. To use this property, you must also define the <code>platform.openstack.externalNetwork</code> property.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>platform.openstack.externalDNS</td>
<td>IP addresses for external DNS servers that cluster instances use for DNS resolution.</td>
<td>A list of IP addresses as strings. For example, [&quot;8.8.8.8&quot;, &quot;192.168.1.12&quot;].</td>
</tr>
<tr>
<td>platform.openstack.machinesSubnet</td>
<td>The UUID of a RHOSP subnet that the cluster’s nodes use. Nodes and virtual IP (VIP) ports are created on this subnet. The first item in <code>networking.machineNetwork</code> must match the value of <code>machinesSubnet</code>. If you deploy to a custom subnet, you cannot specify an external DNS server to the OpenShift Container Platform installer. Instead, add DNS to the subnet in RHOSP.</td>
<td>A UUID as a string. For example, fa806b2f-ac49-4bce-b9db-124bc64209bf.</td>
</tr>
</tbody>
</table>

### 9.5.7.3. Sample customized `install-config.yaml` file for restricted OpenStack installations

This sample `install-config.yaml` demonstrates all of the possible Red Hat OpenStack Platform (RHOSP) customization options.

```yaml
apiVersion: v1
baseDomain: example.com
clusterID: os-test
template:
  name: master
  platform: {}
  replicas: 3
  compute:
    - name: worker
      platform:
        openstack:
```

**IMPORTANT**

This sample file is provided for reference only. You must obtain your `install-config.yaml` file by using the installation program.
9.5.8. Setting compute machine affinity

Optionally, you can set the affinity policy for compute machines during installation. The installer does not select an affinity policy for compute machines by default.

You can also create machine sets that use particular RHOSP server groups after installation.

```
---
type: ml.large
replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
    - hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  serviceNetwork:
    - 172.30.0.0/16
  networkType: OpenShiftSDN
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    lbFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle:
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ ZZ

9.5.8. Setting compute machine affinity

Optionally, you can set the affinity policy for compute machines during installation. The installer does not select an affinity policy for compute machines by default.

You can also create machine sets that use particular RHOSP server groups after installation.

**NOTE**

Control plane machines are created with a **soft-anti-affinity** policy.

**TIP**

You can learn more about *RHOSP instance scheduling and placement* in the RHOSP documentation.

**Prerequisites**
• Create the **install-config.yaml** file and complete any modifications to it.

**Procedure**

1. Using the RHOSP command-line interface, create a server group for your compute machines. For example:

   ```bash
   $ openstack \
   --os-compute-api-version=2.15 \
   server group create \
   --policy anti-affinity \
   my-openshift-worker-group
   ```

   For more information, see the `server group create` command documentation.

2. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir=<installation_directory>
   ```

   where:

   **installation_directory**
   - Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

3. Open `manifests/99_openshift-cluster-api_worker-machineset-0.yaml`, the **MachineSet** definition file.

4. Add the property `serverGroupID` to the definition beneath the `spec.template.spec.providerSpec.value` property. For example:

   ```yaml
   apiVersion: machine.openshift.io/v1beta1
   kind: MachineSet
   metadata:
   labels:
     machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
     machine.openshift.io/cluster-api-machine-role: <node_role>
     machine.openshift.io/cluster-api-machine-type: <node_role>
     name: <infrastructure_ID>-<node_role>
     namespace: openshift-machine-api
   spec:
   replicas: <number_of_replicas>
   selector:
     matchLabels:
     machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
     machine.openshift.io/cluster-api-machine-role: <node_role>
     machine.openshift.io/cluster-api-machineset: <infrastructure_ID>-<node_role>
   template:
   metadata:
   labels:
     machine.openshift.io/cluster-api-cluster: <infrastructure_ID>
     machine.openshift.io/cluster-api-machine-role: <node_role>
     machine.openshift.io/cluster-api-machine-type: <node_role>
   spec:
   providerSpec:
   ```
Add the UUID of your server group here.

5. Optional: Back up the manifests/99_openshift-cluster-api_worker-machineset-0.yaml file. The installation program deletes the manifests/ directory when creating the cluster.

When you install the cluster, the installer uses the MachineSet definition that you modified to create compute machines within your RHOSP server group.

9.5.9. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "" -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$\$(ssh-agent -s)"
   ```

   **Example output**

   
   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.
9.5.10. Enabling access to the environment

At deployment, all OpenShift Container Platform machines are created in a Red Hat OpenStack Platform (RHOSP)-tenant network. Therefore, they are not accessible directly in most RHOSP deployments.

You can configure OpenShift Container Platform API and application access by using floating IP addresses (FIPs) during installation. You can also complete an installation without configuring FIPs, but the installer will not configure a way to reach the API or applications externally.

9.5.10.1. Enabling access with floating IP addresses

Create floating IP (FIP) addresses for external access to the OpenShift Container Platform API and cluster applications.

Procedure

1. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the API FIP:

   ```bash
   $ openstack floating ip create --description "API <cluster_name>.<base_domain>" <external_network>
   ```

2. Using the Red Hat OpenStack Platform (RHOSP) CLI, create the apps, or Ingress, FIP:

   ```bash
   $ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>" <external_network>
   ```

3. Add records that follow these patterns to your DNS server for the API and Ingress FIPs:

   ```
   api.<cluster_name>.<base_domain>. IN A <API_FIP>
   *.apps.<cluster_name>.<base_domain>. IN A <apps_FIP>
   ```

   **NOTE**

   If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

4. Add the FIPs to the `install-config.yaml` file as the values of the following parameters:

   - `platform.openstack.ingressFloatingIP`
   - `platform.openstack.lbFloatingIP`

   If you use these values, you must also enter an external network as the value of the `platform.openstack.externalNetwork` parameter in the `install-config.yaml` file.

**TIP**

You can make OpenShift Container Platform resources available outside of the cluster by assigning a floating IP address and updating your firewall configuration.

9.5.10.2. Completing installation without floating IP addresses

1566
You can install OpenShift Container Platform on Red Hat OpenStack Platform (RHOSP) without providing floating IP addresses.

In the `install-config.yaml` file, do not define the following parameters:

- `platform.openstack.ingressFloatingIP`
- `platform.openstack.lbFloatingIP`

If you cannot provide an external network, you can also leave `platform.openstack.externalNetwork` blank. If you do not provide a value for `platform.openstack.externalNetwork`, a router is not created for you, and, without additional action, the installer will fail to retrieve an image from Glance. You must configure external connectivity on your own.

If you run the installer from a system that cannot reach the cluster API due to a lack of floating IP addresses or name resolution, installation fails. To prevent installation failure in these cases, you can use a proxy network or run the installer from a system that is on the same network as your machines.

**NOTE**

You can enable name resolution by creating DNS records for the API and Ingress ports. For example:

```shell
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

If you do not control the DNS server, you can add the record to your `/etc/hosts` file. This action makes the API accessible to only you, which is not suitable for production deployment but does allow installation for development and testing.

### 9.5.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
   ```
For `<installation_directory>`, specify the

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```bash
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-yMBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.
9.5.12. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

Procedure

1. In the cluster environment, export the administrator’s kubeconfig file:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

   $ oc get nodes

3. View your cluster’s version:

   $ oc get clusterversion

4. View your Operators’ status:

   $ oc get clusteroperator

5. View all running pods in the cluster:

   $ oc get pods -A

9.5.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig

   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   
   Example output
   
   system:admin
   ```

   Additional resources

   - See [Accessing the web console](#) for more details about accessing and understanding the OpenShift Container Platform web console.

9.5.14. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```
  $ oc patch OperatorHub cluster --type json -p '[_("op": "add", "path": "/spec/disableAllDefaultSources", "value": true)]'
  ```

TIP

Alternatively, you can use the web console to manage catalog sources. From the `Administration → Cluster Settings → Global Configuration → OperatorHub` page, click the `Sources` tab, where you can create, delete, disable, and enable individual sources.

9.5.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See [About remote health monitoring](#) for more information about the Telemetry service

9.5.16. Next steps
- Customize your cluster.

- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.

- If necessary, you can opt out of remote health reporting.

- Configure image streams for the Cluster Samples Operator and the `must-gather` tool.

- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.

- If you did not configure RHOSP to accept application traffic over floating IP addresses, configure RHOSP access with floating IP addresses.

9.6. UNINSTALLING A CLUSTER ON OPENSTACK

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP).

9.6.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

**NOTE**

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

**Prerequisites**

- Have a copy of the installation program that you used to deploy the cluster.

- Have the files that the installation program generated when you created your cluster.

**Procedure**

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

   ```
   $ ./openshift-install destroy cluster \
   --dir <installation_directory> --log-level info
   ```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.
9.7. UNINSTALLING A CLUSTER ON RHOSP FROM YOUR OWN INFRASTRUCTURE

You can remove a cluster that you deployed to Red Hat OpenStack Platform (RHOSP) on user-provisioned infrastructure.

9.7.1. Downloading playbook dependencies

The Ansible playbooks that simplify the removal process on user-provisioned infrastructure require several Python modules. On the machine where you will run the process, add the modules' repositories and then download them.

NOTE

These instructions assume that you are using Red Hat Enterprise Linux (RHEL) 8.

Prerequisites

- Python 3 is installed on your machine.

Procedure

1. On a command line, add the repositories:

   a. Register with Red Hat Subscription Manager:

      ```
      $ sudo subscription-manager register # If not done already
      ```

   b. Pull the latest subscription data:

      ```
      $ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
      ```

   c. Disable the current repositories:

      ```
      $ sudo subscription-manager repos --disable=* # If not done already
      ```

   d. Add the required repositories:

      ```
      $ sudo subscription-manager repos \ 
      --enable=rhel-8-for-x86_64-baseos-rpms \ 
      --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \ 
      --enable=ansible-2.9-for-rhel-8-x86_64-rpms \ 
      --enable=rhel-8-for-x86_64-appstream-rpms
      ```

2. Install the modules:

   ```
   $ sudo yum install python3-openstackclient ansible python3-openstacksdk
   ```

3. Ensure that the **python** command points to **python3**:
9.7.2. Removing a cluster from RHOSP that uses your own infrastructure

You can remove an OpenShift Container Platform cluster on Red Hat OpenStack Platform (RHOSP) that uses your own infrastructure. To complete the removal process quickly, run several Ansible playbooks.

Prerequisites

- Python 3 is installed on your machine.
- You downloaded the modules in "Downloading playbook dependencies."
- You have the playbooks that you used to install the cluster.
- You modified the playbooks that are prefixed with down- to reflect any changes that you made to their corresponding installation playbooks. For example, changes to the bootstrap.yaml file are reflected in the down-bootstrap.yaml file.
- All of the playbooks are in a common directory.

Procedure

1. On a command line, run the playbooks that you downloaded:

   ```bash
   $ sudo alternatives --set python /usr/bin/python3
   $ ansible-playbook -i inventory.yaml down-bootstrap.yaml down-control-plane.yaml down-compute-nodes.yaml down-load-balancers.yaml down-network.yaml down-security-groups.yaml
   ```

2. Remove any DNS record changes you made for the OpenShift Container Platform installation.

OpenShift Container Platform is removed from your infrastructure.
CHAPTER 10. INSTALLING ON RHV

10.1. INSTALLING A CLUSTER QUICKLY ON RHV

You can quickly install a default, non-customized, OpenShift Container Platform cluster on a Red Hat Virtualization (RHV) cluster, similar to the one shown in the following diagram.

The installation program uses installer-provisioned infrastructure to automate creating and deploying the cluster.

To install a default cluster, you prepare the environment, run the installation program and answer its prompts. Then, the installation program creates the OpenShift Container Platform cluster.

For an alternative to installing a default cluster, see Installing a cluster with customizations.

NOTE

This installation program is available for Linux and macOS only.

10.1.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- You have a supported combination of versions in the Support Matrix for OpenShift Container Platform on Red Hat Virtualization (RHV).
- If you use a firewall, configure it to allow the sites that your cluster requires access to.
10.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

10.1.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on default values multiplied by the default number of virtual machines the installation program creates. These resources must be available in addition to what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

**Requirements**

- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
  - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
- 112 GiB RAM or more, including:
  - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
  - 16 GiB or more for each of the three control plane machines which provide the control plane.
  - 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet these etcd backend performance requirements.
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
  - DiskOperator
  - DiskCreator
  - UserTemplateBasedVm
  - TemplateOwner
  - TemplateCreator
  - ClusterAdmin on the target cluster

WARNING

Apply the principle of least privilege: Avoid using an administrator account with SuperUser privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary ovirt-config.yaml file that might be compromised.

10.1.4. Verifying the requirements for the RHV environment

Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.
IMPORTANT

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check the RHV version.
   a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select About.
   b. In the window that opens, make a note of the RHV Software Version.
   c. Confirm that version 4.6 of OpenShift Container Platform and the version of RHV you noted are one of the supported combinations in the Support Matrix for OpenShift Container Platform on RHV.

2. Inspect the data center, cluster, and storage.
   a. In the RHV Administration Portal, click Compute → Data Centers.
   b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
   c. Click the name of that data center.
   d. In the data center details, on the Storage tab, confirm the storage domain where you plan to install OpenShift Container Platform is Active.
   e. Record the Domain Name for use later on.
   f. Confirm Free Space has at least 230 GiB.
   g. Confirm that the storage domain meets these etcd backend performance requirements, which you can measure by using the fio performance benchmarking tool.
   h. In the data center details, click the Clusters tab.
   i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.

3. Inspect the RHV host resources.
   a. In the RHV Administration Portal, click Compute > Clusters.
   b. Click the cluster where you plan to install OpenShift Container Platform.
   c. In the cluster details, click the Hosts tab.
   d. Inspect the hosts and confirm they have a combined total of at least 28 Logical CPU Cores available exclusively for the OpenShift Container Platform cluster.
   e. Record the number of available Logical CPU Cores for use later on.
f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.

g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:

- 16 GiB required for the bootstrap machine
- 16 GiB required for each of the three control plane machines
- 16 GiB for each of the three compute machines

h. Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.

4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager’s REST API. From a virtual machine on this network, use curl to reach the RHV Manager’s REST API:

```
$ curl -k -u <username>@<profile>:<password> \ 
    https://<engine-fqdn>/ovirt-engine/api
```

For `<username>`, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For `<profile>`, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. For `<password>`, specify the password for that user name.

```
$ curl -k -u ocpadmin@internal:pw123 \ 
    https://rhv-env.virtlab.example.com/ovirt-engine/api
```

### 10.1.5. Preparing the network environment on RHV

Configure two static IP addresses for the OpenShift Container Platform cluster and create DNS entries using these addresses.

**Procedure**

1. Reserve two static IP addresses
   
   a. On the network where you plan to install OpenShift Container Platform, identify two static IP addresses that are outside the DHCP lease pool.
   
   b. Connect to a host on this network and verify that each of the IP addresses is not in use. For example, use Address Resolution Protocol (ARP) to check that none of the IP addresses have entries:

```
$ arp 10.35.1.19
```
Example output

| 10.35.1.19 (10.35.1.19) -- no entry |

c. Reserve two static IP addresses following the standard practices for your network environment.

d. Record these IP addresses for future reference.

2. Create DNS entries for the OpenShift Container Platform REST API and apps domain names using this format:

   api.<cluster-name>.<base-domain>  <ip-address>  1
   *.apps.<cluster-name>.<base-domain>  <ip-address>  2

   For <cluster-name>, <base-domain>, and <ip-address>, specify the cluster name, base domain, and static IP address of your OpenShift Container Platform API.

   Specify the cluster name, base domain, and static IP address of your OpenShift Container Platform apps for Ingress and the load balancer.

   For example:

   | api.my-cluster.virtlab.example.com 10.35.1.19 |
   | *.apps.my-cluster.virtlab.example.com 10.35.1.20 |

10.1.6. Setting up the CA certificate for RHV

Download the CA certificate from the Red Hat Virtualization (RHV) Manager and set it up on the installation machine.

You can download the certificate from a webpage on the RHV Manager or by using a curl command.

Later, you provide the certificate to the installation program.

Procedure

1. Use either of these two methods to download the CA certificate:

   - Go to the Manager’s webpage, https://<engine-fqdn>/ovirt-engine/. Then, under Downloads, click the CA Certificate link.
   
   - Run the following command:

     ```
     $ curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem 1
     ```

     For <engine-fqdn>, specify the fully qualified domain name of the RHV Manager, such as rhv-env.virtlab.example.com.

2. Configure the CA file to grant rootless user access to the Manager. Set the CA file permissions to have an octal value of **0644** (symbolic value: -rwxr--r--):
3. For Linux, copy the CA certificate to the directory for server certificates. Use `-p` to preserve the permissions:

```
$ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
```

4. Add the certificate to the certificate manager for your operating system:

   - For macOS, double-click the certificate file and use the **Keychain Access** utility to add the file to the **System** keychain.
   - For Linux, update the CA trust:

```
$ sudo update-ca-trust
```

**NOTE**

If you use your own certificate authority, make sure the system trusts it.

**Additional resources**

- To learn more, see **Authentication and Security** in the RHV documentation.

### 10.1.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your **ssh-agent** and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user **core**. When you deploy the cluster, the key is added to the **core** user’s `~/.ssh/authorized_keys` list.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N " \\
-f <path>/<file_name>
```

Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.
NOTE

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   $ eval "$(ssh-agent -s)"

Example output

   Agent pid 31874

NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   $ ssh-add <path>/<file_name> 1

Example output

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.1.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.
3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

10.1.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the create cluster command of the installation program only once, during initial installation.

**Prerequisites**

- Open the ovirt-imageio port to the Manager from the machine running the installer. By default, the port is 54322.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
```

1582
1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Respond to the installation program prompts.

1. **Optional:** For **SSH Public Key**, select a password-less public key, such as `~/.ssh/id_rsa.pub`. This key authenticates connections with the new OpenShift Container Platform cluster.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, select an SSH key that your `ssh-agent` process uses.

2. For **Platform**, select `ovirt`.

3. For **Engine FQDN[:PORT]**, enter the fully qualified domain name (FQDN) of the RHV environment.
   For example:
   ```
   rhv-env.virtlab.example.com:443
   ```

4. The installer automatically generates a CA certificate. For **Would you like to use the above certificate to connect to the Manager?**, answer *y* or *N*. If you answer *N*, you must install OpenShift Container Platform in insecure mode.

5. For **Engine username**, enter the user name and profile of the RHV administrator using this format:
   ```
   <username>[@<profile>]
   ```
   For `<username>`, specify the user name of an RHV administrator. For `<profile>`, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the Profile dropdown list. For example: `admin@internal`.

6. For **Engine password**, enter the RHV admin password.

7. For **Cluster**, select the RHV cluster for installing OpenShift Container Platform.
h. For **Storage domain**, select the storage domain for installing OpenShift Container Platform.

i. For **Network**, select a virtual network that has access to the RHV Manager REST API.

j. For **Internal API Virtual IP**, enter the static IP address you set aside for the cluster’s REST API.

k. For **Ingress virtual IP**, enter the static IP address you reserved for the wildcard apps domain.

l. For **Base Domain**, enter the base domain of the OpenShift Container Platform cluster. If this cluster is exposed to the outside world, this must be a valid domain recognized by DNS infrastructure. For example, enter: `virtlab.example.com`

m. For **Cluster Name**, enter the name of the cluster. For example, `my-cluster`. Use cluster name from the externally registered/resolvable DNS entries you created for the OpenShift Container Platform REST API and apps domain names. The installation program also gives this name to the cluster in the RHV environment.

n. For **Pull Secret**, copy the pull secret from the `pull-secret.txt` file you downloaded earlier and paste it here. You can also get a copy of the same pull secret from the Red Hat OpenShift Cluster Manager.

---

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

---

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

IMPORTANT

You have completed the steps required to install the cluster. The remaining steps show you how to verify the cluster and troubleshoot the installation.

10.1.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

10.1.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   `$ tar xvzf <file>`
5. Place the `oc` binary in a directory that is on your PATH. To check your PATH, execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 10.1.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

```
C:\> path
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 10.1.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

To learn more, see Getting started with the OpenShift CLI.

### 10.1.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

#### Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

#### Procedure

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   ```
   system:admin
   ```

#### Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

### 10.1.12. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

#### Procedure

1. In the cluster environment, export the administrator’s `kubeconfig` file:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:
   
   $ oc get nodes

3. View your cluster's version:
   
   $ oc get clusterversion

4. View your Operators' status:
   
   $ oc get clusteroperator

5. View all running pods in the cluster:
   
   $ oc get pods -A

Troubleshooting

If the installation fails, the installation program times out and displays an error message. To learn more, see [Troubleshooting installation issues](#).

10.1.13. Accessing the OpenShift Container Platform web console on RHV

After the OpenShift Container Platform cluster initializes, you can log in to the OpenShift Container Platform web console.

**Procedure**

1. Optional: In the Red Hat Virtualization (RHV) Administration Portal, open **Compute → Cluster**.

2. Verify that the installation program creates the virtual machines.

3. Return to the command line where the installation program is running. When the installation program finishes, it displays the user name and temporary password for logging into the OpenShift Container Platform web console.

4. In a browser, open the URL of the OpenShift Container Platform web console. The URL uses this format:

   console-openshift-console.apps.<clusternamex>.<basedomainx>

   For `<clusternamex>.<basedomainx>`, specify the cluster name and base domain.

   For example:

   console-openshift-console.apps.my-cluster.virtlab.example.com
10.1.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

10.1.15. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)

Here are some common issues you might encounter, along with proposed causes and solutions.

10.1.15.1. CPU load increases and nodes go into a Not Ready state

- **Symptom**: CPU load increases significantly and nodes start going into a Not Ready state.

- **Cause**: The storage domain latency might be too high, especially for control plane nodes (also known as the master nodes).

- **Solution**:
  Make the nodes ready again by restarting the kubelet service:

  ```
  $ systemctl restart kubelet
  ```

  Inspect the OpenShift Container Platform metrics service, which automatically gathers and reports on some valuable data such as the etcd disk sync duration. If the cluster is operational, use this data to help determine whether storage latency or throughput is the root issue. If so, consider using a storage resource that has lower latency and higher throughput.

  To get raw metrics, enter the following command as kubeadmin or user with cluster-admin privileges:

  ```
  ```

  To learn more, see Exploring Application Endpoints for the purposes of Debugging with OpenShift 4.x

10.1.15.2. Trouble connecting the OpenShift Container Platform cluster API

- **Symptom**: The installation program completes but the OpenShift Container Platform cluster API is not available. The bootstrap virtual machine remains up after the bootstrap process is complete. When you enter the following command, the response will time out.

  ```
  $ oc login -u kubeadmin -p *** <apiurl>
  ```

- **Cause**: The bootstrap VM was not deleted by the installation program and has not released the cluster’s API IP address.
Solution: Use the `wait-for` subcommand to be notified when the bootstrap process is complete:

```bash
$ ./openshift-install wait-for bootstrap-complete
```

When the bootstrap process is complete, delete the bootstrap virtual machine:

```bash
$ ./openshift-install destroy bootstrap
```

10.1.16. Post-installation tasks

After the OpenShift Container Platform cluster initializes, you can perform the following tasks.

- Optional: After deployment, add or replace SSH keys using the Machine Config Operator (MCO) in OpenShift Container Platform.

- Optional: Remove the `kubeadmin` user. Instead, use the authentication provider to create a user with `cluster-admin` privileges.

10.2. INSTALLING A CLUSTER ON RHV WITH CUSTOMIZATIONS

You can customize and install an OpenShift Container Platform cluster on Red Hat Virtualization (RHV), similar to the one shown in the following diagram.

The installation program uses installer-provisioned infrastructure to automate creating and deploying the cluster.

To install a customized cluster, you prepare the environment and perform the following steps:
1. Create an installation configuration file, the `install-config.yaml` file, by running the installation program and answering its prompts.

2. Inspect and modify parameters in the `install-config.yaml` file.

3. Make a working copy of the `install-config.yaml` file.

4. Run the installation program with a copy of the `install-config.yaml` file.

Then, the installation program creates the OpenShift Container Platform cluster.

For an alternative to installing a customized cluster, see Installing a default cluster.

**NOTE**

This installation program is available for Linux and macOS only.

### 10.2.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.

- You have a supported combination of versions in the Support Matrix for OpenShift Container Platform on Red Hat Virtualization (RHV).

- If you use a firewall, configure it to allow the sites that your cluster requires access to.

### 10.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 10.2.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform cluster, the RHV environment must meet the following requirements.
Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on default values multiplied by the default number of virtual machines the installation program creates. These resources must be available in addition to what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

Requirements

- The RHV environment has one data center whose state is Up.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
  - Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.
  - 112 GiB RAM or more, including:
    - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
    - 16 GiB or more for each of the three control plane machines which provide the control plane.
    - 16 GiB or more for each of the three compute machines, which run the application workloads.
- The RHV storage domain must meet these etcd backend performance requirements.
- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.
- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.
- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.
- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
DiskOperator
DiskCreator
UserTemplateBasedVm
TemplateOwner
TemplateCreator
ClusterAdmin on the target cluster

WARNING
Apply the principle of least privilege: Avoid using an administrator account with SuperUser privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary ovirt-config.yaml file that might be compromised.

10.2.4. Verifying the requirements for the RHV environment
Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.

IMPORTANT
These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

Procedure

1. Check the RHV version.
   a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select About.
   b. In the window that opens, make a note of the RHV Software Version
   c. Confirm that version 4.6 of OpenShift Container Platform and the version of RHV you noted are one of the supported combinations in the Support Matrix for OpenShift Container Platform on RHV.

2. Inspect the data center, cluster, and storage.
   a. In the RHV Administration Portal, click Compute → Data Centers.
   b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
c. Click the name of that data center.

d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.

e. Record the **Domain Name** for use later on.

f. Confirm **Free Space** has at least 230 GiB.

g. Confirm that the storage domain meets [these etcd backend performance requirements](#), which you can measure by using the **fio** performance benchmarking tool.

h. In the data center details, click the **Clusters** tab.

i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.

3. Inspect the RHV host resources.

a. In the RHV Administration Portal, click **Compute > Clusters**.

b. Click the cluster where you plan to install OpenShift Container Platform.

c. In the cluster details, click the **Hosts** tab.

d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available **exclusively** for the OpenShift Container Platform cluster.

e. Record the number of available **Logical CPU Cores** for use later on.

f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.

g. Confirm that, all together, the hosts have 112 GiB of **Max free Memory for scheduling new virtual machines** distributed to meet the requirements for each of the following OpenShift Container Platform machines:

   - 16 GiB required for the bootstrap machine
   - 16 GiB required for each of the three control plane machines
   - 16 GiB for each of the three compute machines

h. Record the amount of **Max free Memory for scheduling new virtual machines** for use later on.

4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager’s REST API. From a virtual machine on this network, use curl to reach the RHV Manager’s REST API:

   ```bash
   $ curl -k -u <username>@<profile>:<password> \                   
    https://<engine-fqdn>/ovirt-engine/api
   ```

   **For** `<username>`, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. **For** `<profile>`, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. **For** `<password>`, specify the password for that user.
name.

2. For `<engine-fqdn>`, specify the fully qualified domain name of the RHV environment.

For example:

```
$ curl -k -u ocpadmin@internal:pw123 \
https://rhv-env.virtlab.example.com/ovirt-engine/api
```

### 10.2.5. Preparing the network environment on RHV

Configure two static IP addresses for the OpenShift Container Platform cluster and create DNS entries using these addresses.

**Procedure**

1. Reserve two static IP addresses

   a. On the network where you plan to install OpenShift Container Platform, identify two static IP addresses that are outside the DHCP lease pool.

   b. Connect to a host on this network and verify that each of the IP addresses is not in use. For example, use Address Resolution Protocol (ARP) to check that none of the IP addresses have entries:

   ```
   $ arp 10.35.1.19
   ```

   **Example output**

   ```
   10.35.1.19 (10.35.1.19) -- no entry
   ```

   c. Reserve two static IP addresses following the standard practices for your network environment.

   d. Record these IP addresses for future reference.

2. Create DNS entries for the OpenShift Container Platform REST API and apps domain names using this format:

   ```
   api.<cluster-name>.<base-domain>   <ip-address> 1
   *.apps.<cluster-name>.<base-domain>   <ip-address> 2
   ```

   For `<cluster-name>`, `<base-domain>`, and `<ip-address>`, specify the cluster name, base domain, and static IP address of your OpenShift Container Platform API.

   Specify the cluster name, base domain, and static IP address of your OpenShift Container Platform apps for Ingress and the load balancer.

   For example:

   ```
   api.my-cluster.virtlab.example.com 10.35.1.19
   *.apps.my-cluster.virtlab.example.com 10.35.1.20
   ```
10.2.6. Setting up the CA certificate for RHV

Download the CA certificate from the Red Hat Virtualization (RHV) Manager and set it up on the installation machine.

You can download the certificate from a webpage on the RHV Manager or by using a `curl` command. Later, you provide the certificate to the installation program.

Procedure

1. Use either of these two methods to download the CA certificate:
   - Go to the Manager’s webpage, [https://<engine-fqdn>/ovirt-engine/](https://<engine-fqdn>/ovirt-engine/). Then, under **Downloads**, click the **CA Certificate** link.
   - Run the following command:
     ```bash
     ```
     For `<engine-fqdn>`, specify the fully qualified domain name of the RHV Manager, such as `rhv-env.virtlab.example.com`.

2. Configure the CA file to grant rootless user access to the Manager. Set the CA file permissions to have an octal value of **0644** (symbolic value: `-rw-r--r--`):
   ```bash
   $ sudo chmod 0644 /tmp/ca.pem
   ```

3. For Linux, copy the CA certificate to the directory for server certificates. Use `-p` to preserve the permissions:
   ```bash
   $ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
   $$ sudo update-ca-trust
   ```

4. Add the certificate to the certificate manager for your operating system:
   - For macOS, double-click the certificate file and use the **Keychain Access** utility to add the file to the **System** keychain.
   - For Linux, update the CA trust:
     ```bash
     $ sudo update-ca-trust
     ```

   **NOTE**
   If you use your own certificate authority, make sure the system trusts it.

Additional resources

- To learn more, see **Authentication and Security** in the RHV documentation.

10.2.7. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N " "
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**
Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.2.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.
10.2.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Red Hat Virtualization (RHV).

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>  
   ```

   1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

   IMPORTANT

   Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. Respond to the installation program prompts.

      i. For **SSH Public Key**, select a password-less public key, such as `~/.ssh/id_rsa.pub`. This key authenticates connections with the new OpenShift Container Platform cluster.

         NOTE

         For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, select an SSH key that your `ssh-agent` process uses.

      ii. For **Platform**, select **ovirt**.

      iii. For **Enter oVirt’s API endpoint URL**, enter the URL of the RHV API using this format:

         ```
         https://<engine-fqdn>/ovirt-engine/api  
         ```

         1 For `<engine-fqdn>`, specify the fully qualified domain name of the RHV environment.
For example:

```
$ curl -k -u ocpadmin@internal:pw123 \\
hits/rhv-env.virtlab.example.com/ovirt-engine/api
```

iv. For **Is the oVirt CA trusted locally?**, enter **Yes** since you have already set up a CA certificate. Otherwise, enter **No**.

v. For **oVirt’s CA bundle**, if you entered **Yes** for the preceding question, copy the certificate content from `/etc/pki/ca-trust/source/anchors/ca.pem` and paste it here. Then, press **Enter** twice. Otherwise, if you entered **No** for the preceding question, this question does not appear.

vi. For **oVirt engine username**, enter the user name and profile of the RHV administrator using this format:

```
<username>@<profile>
```

For **<username>**, specify the user name of an RHV administrator. For **<profile>**, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the **Profile** dropdown list. Together, the user name and profile should look similar to this example:

```
ocpadmin@internal
```

vii. For **oVirt engine password**, enter the RHV admin password.

viii. For **oVirt cluster**, select the cluster for installing OpenShift Container Platform.

ix. For **oVirt storage domain**, select the storage domain for installing OpenShift Container Platform.

x. For **oVirt network**, select a virtual network that has access to the Manager REST API.

xi. For **Internal API Virtual IP**, enter the static IP address you set aside for the cluster’s REST API.

xii. For **Ingress virtual IP**, enter the static IP address you reserved for the wildcard apps domain.

xiii. For **Base Domain**, enter the base domain of the OpenShift Container Platform cluster. If this cluster is exposed to the outside world, this must be a valid domain recognized by DNS infrastructure. For example, enter: `virtlab.example.com`

xiv. For **Cluster Name**, enter the name of the cluster. For example, **my-cluster**. Use cluster name from the externally registered/resolvable DNS entries you created for the OpenShift Container Platform REST API and apps domain names. The installation program also gives this name to the cluster in the RHV environment.

xv. For **Pull Secret**, copy the pull secret from the **pull-secret.txt** file you downloaded earlier and paste it here. You can also get a copy of the same pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the **install-config.yaml** file. You can find more information about the available parameters in the **Installation configuration parameters** section.
3. Back up the install-config.yaml file so that you can use it to install multiple clusters.

**IMPORTANT**

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 10.2.9.1. Example install-config.yaml files for Red Hat Virtualization (RHV)

You can customize the OpenShift Container Platform cluster the installation program creates by changing the parameters and parameter values in the install-config.yaml file.

The following example is specific to installing OpenShift Container Platform on RHV.

This file is located in the <installation_directory> you specified when you ran the following command.

```
$ ./openshift-install create install-config --dir <installation_directory>
```

**NOTE**

- These example files are provided for reference only. You must obtain your install-config.yaml file by using the installation program.

- Changing the install-config.yaml file can increase the resources your cluster requires. Verify that your RHV environment has those additional resources. Otherwise, the installation or cluster will fail.

**Example: This is the default install-config.yaml file**

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - architecture: amd64
    hypervthreading: Enabled
    name: worker
    platform: {}
    replicas: 3
controlPlane:
  architecture: amd64
  hypervthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: my-cluster
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
networkType: OpenShiftSDN
serviceNetwork:
```
platform:
  api_vip: 10.46.8.230
  ingress_vip: 192.168.1.5
  ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
  ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
  ovirt_network_name: ovirtmgmt
  vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...

Example: A minimal install-config.yaml file

apiVersion: v1
baseDomain: example.com
metadata:
  name: test-cluster
platform:
  ovirt:
    api_vip: 10.46.8.230
    ingress_vip: 10.46.8.232
    ovirt_cluster_id: 68833f9f-e89c-4891-b768-e2ba0815b76b
    ovirt_storage_domain_id: ed7b0f4e-0e96-492a-8fff-279213ee1468
    ovirt_network_name: ovirtmgmt
    vnicProfileID: 3fa86930-0be5-4052-b667-b79f0a729692
pullSecret: '{"auths": ...}'
sshKey: ssh-ed12345 AAAA...

Example: Custom machine pools in an install-config.yaml file

apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
platform:
  ovirt:
    cpu:
      cores: 4
      sockets: 2
      memoryMB: 65536
  osDisk:
    sizeGB: 100
    vmType: server
  replicas: 3
compute:
  - name: worker
    platform:
      ovirt:
        cpu:
          cores: 4
          sockets: 4
          memoryMB: 65536
          osDisk:
10.2.9.2. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

10.2.9.2.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 10.1. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is V1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}.{{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{ &quot;auths&quot;:{ &quot;cloud.openshift.com&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;you@example.com&quot; }, &quot;quay.io&quot;:{ &quot;auth&quot;:&quot;b3Blb=&quot;, &quot;email&quot;:&quot;you@example.com&quot; } } }</code></td>
</tr>
</tbody>
</table>
### 10.2.9.2.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

#### Table 10.2. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>networking.network</strong></td>
<td>The cluster network provider</td>
<td>Either <code>OpenShiftSDN</code> or <code>OVNKubernetes</code>. The default value is <code>OpenShiftSDN</code>.</td>
</tr>
<tr>
<td>Type</td>
<td>Container Network Interface (CNI) plug-in to install.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>The default value is <code>10.128.0.0/14</code> with a host prefix of <code>/23</code>.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given <code>cidr</code>. A <code>hostPrefix</code> value of 23 provides 510 (2^32 - 32 - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
</tbody>
</table>
### networking.serviceNetwork

The IP address block for services. The default value is `172.30.0.0/16`.

The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.

An array with an IP address block in CIDR format. For example:

```yaml
networking:
  serviceNetwork:
    - 172.30.0.0/16
```

### networking.machineNetwork

The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

An array of objects. For example:

```yaml
networking:
  machineNetwork:
  - cidr: 10.0.0.0/16
```

### networking.machineNetwork.cidr

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

---

### 10.2.9.2.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 10.3. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>IMPORTANT</td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
</tbody>
</table>
## Parameter Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>controlPlane.architecture</strong></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td>String</td>
</tr>
<tr>
<td><strong>controlPlane.hypertreading</strong></td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hypertreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td><strong>controlPlane.name</strong></td>
<td>Required if you use <strong>controlPlane</strong>. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td><strong>controlPlane.platform</strong></td>
<td>Required if you use <strong>controlPlane</strong>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <strong>compute.platform</strong> parameter value.</td>
<td><strong>aws</strong>, <strong>azure</strong>, <strong>gcp</strong>, <strong>openstack</strong>, <strong>ovirt</strong>, <strong>vsphere</strong>, or {}</td>
</tr>
<tr>
<td><strong>controlPlane.replicas</strong></td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
</tbody>
</table>
### credentialsMode

The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td><code>Internal</code> or <code>External</code>. The default value is <code>External</code>. Setting this field to <code>Internal</code> is not supported on non-cloud platforms.</td>
</tr>
<tr>
<td>sshKey</td>
<td>The SSH key or keys to authenticate access your cluster machines.</td>
<td>One or more keys. For example:</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td><code>sshKey: &lt;key1&gt;</code>&lt;key2&gt;&lt;key3&gt;`</td>
</tr>
<tr>
<td></td>
<td>For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your <code>ssh-agent</code> process uses.</td>
<td></td>
</tr>
</tbody>
</table>

### 10.2.9.2.4. Additional Red Hat Virtualization (RHV) configuration parameters

Additional RHV configuration parameters are described in the following table:

**Table 10.4. Additional RHV parameters for clusters**
### Additional RHV parameters for machine pools

Additional RHV configuration parameters for machine pools are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.cpu</code></td>
<td>Optional. Defines the CPU of the VM.</td>
<td>Object</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.cpu.cores</code></td>
<td>Required if you use <code>&lt;machine-pool&gt;.platform.ovirt.cpu</code>. The number of cores. Total virtual CPUs (vCPUs) is cores * sockets.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.cpusockets</code></td>
<td>Required if you use <code>&lt;machine-pool&gt;.platform.ovirt.cpu</code>. The number of sockets per core. Total virtual CPUs (vCPUs) is cores * sockets.</td>
<td>Integer</td>
</tr>
</tbody>
</table>
### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.memoryMB</code></td>
<td>Optional. Memory of the VM in MiB.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.instanceTypeID</code></td>
<td>Optional. An instance type UUID, such as 00000009-0009-0009-0009-0000000000f1, which you can get from the https://&lt;engine-fqdn&gt;/ovirt-engine/api/instancetypes endpoint.</td>
<td>String of UUID</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.osDisk</code></td>
<td>Optional. Defines the first and bootable disk of the VM.</td>
<td>String</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.osDisk.sizeGB</code></td>
<td>Required if you use <code>&lt;machine-pool&gt;.platform.ovirt.osDisk</code>. Size of the disk in GiB.</td>
<td>Number</td>
</tr>
<tr>
<td><code>&lt;machine-pool&gt;.platform.ovirt.vmType</code></td>
<td>Optional. The VM workload type, such as high-performance, server, or desktop.</td>
<td>String</td>
</tr>
</tbody>
</table>

**NOTE**

You can replace `<machine-pool>` with `controlPlane` or `compute`.

10.2.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Open the `ovirt-imageio` port to the Manager from the machine running the installer. By default, the port is 54322.

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**
1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
```

   1 For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
IMPORTANT
You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

IMPORTANT
You have completed the steps required to install the cluster. The remaining steps show you how to verify the cluster and troubleshoot the installation.

10.2.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

IMPORTANT
If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

10.2.11.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

10.2.11.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the oc command:

   ```
   C:\> oc <command>
   ```

**10.2.11.3. Installing the OpenShift CLI on macOS**

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

**10.2.12. Logging in to the cluster by using the CLI**

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
• You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   ```
   system:admin
   ```

   To learn more, see *Getting started with the OpenShift CLI*.

### 10.2.13. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

**Procedure**

1. In the cluster environment, export the administrator’s kubeconfig file:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

   ```
   $ oc get nodes
   ```

3. View your cluster's version:

   ```
   $ oc get clusterversion
   ```

4. View your Operators' status:

   ```
   $ oc get clusteroperator
   ```

5. View all running pods in the cluster:
Troubleshooting

If the installation fails, the installation program times out and displays an error message. To learn more, see Troubleshooting installation issues.

10.2.14. Accessing the OpenShift Container Platform web console on RHV

After the OpenShift Container Platform cluster initializes, you can log in to the OpenShift Container Platform web console.

Procedure


2. Verify that the installation program creates the virtual machines.

3. Return to the command line where the installation program is running. When the installation program finishes, it displays the user name and temporary password for logging into the OpenShift Container Platform web console.

4. In a browser, open the URL of the OpenShift Container Platform web console. The URL uses this format:

   \[\text{console-openshift-console.apps.<clusternam>.<basedomain}>\]

   For `<clusternam>.<basedomain>`, specify the cluster name and base domain.

   For example:

   \[\text{console-openshift-console.apps.my-cluster.virtlab.example.com}\]

10.2.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

10.2.16. Troubleshooting common issues with installing on Red Hat Virtualization (RHV)

Here are some common issues you might encounter, along with proposed causes and solutions.

10.2.16.1. CPU load increases and nodes go into a Not Ready state
• **Symptom**: CPU load increases significantly and nodes start going into a **Not Ready** state.

• **Cause**: The storage domain latency might be too high, especially for control plane nodes (also known as the master nodes).

• **Solution**:
  Make the nodes ready again by restarting the kubelet service:

  ```
  $ systemctl restart kubelet
  ```

  Inspect the OpenShift Container Platform metrics service, which automatically gathers and reports on some valuable data such as the etcd disk sync duration. If the cluster is operational, use this data to help determine whether storage latency or throughput is the root issue. If so, consider using a storage resource that has lower latency and higher throughput.

  To get raw metrics, enter the following command as kubeadmin or user with cluster-admin privileges:

  ```
  ```

  To learn more, see [Exploring Application Endpoints for the purposes of Debugging with OpenShift 4.x](#).

---

10.2.16.2. **Trouble connecting the OpenShift Container Platform cluster API**

• **Symptom**: The installation program completes but the OpenShift Container Platform cluster API is not available. The bootstrap virtual machine remains up after the bootstrap process is complete. When you enter the following command, the response will time out.

  ```
  $ oc login -u kubeadmin -p *** <apiurl>
  ```

• **Cause**: The bootstrap VM was not deleted by the installation program and has not released the cluster’s API IP address.

• **Solution**: Use the `wait-for` subcommand to be notified when the bootstrap process is complete:

  ```
  $ ./openshift-install wait-for bootstrap-complete
  ```

  When the bootstrap process is complete, delete the bootstrap virtual machine:

  ```
  $ ./openshift-install destroy bootstrap
  ```

---

10.2.17. **Post-installation tasks**

After the OpenShift Container Platform cluster initializes, you can perform the following tasks.

• Optional: After deployment, add or replace SSH keys using the Machine Config Operator (MCO) in OpenShift Container Platform.

• Optional: Remove the `kubeadmin` user. Instead, use the authentication provider to create a user with cluster-admin privileges.

---

10.2.18. **Next steps**
Customize your cluster.

If necessary, you can opt out of remote health reporting.

10.3. INSTALLING A CLUSTER ON RHV WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a customized OpenShift Container Platform cluster on Red Hat Virtualization (RHV) and other infrastructure that you provide. The OpenShift Container Platform documentation uses the term *user-provisioned infrastructure* to refer to this infrastructure type.

The following diagram shows an example of a potential OpenShift Container Platform cluster running on a RHV cluster.

The RHV hosts run virtual machines that contain both control plane and compute pods. One of the hosts also runs a Manager virtual machine and a bootstrap virtual machine that contains a temporary control plane pod.

10.3.1. Prerequisites

The following items are required to install an OpenShift Container Platform cluster on a RHV environment.

- You have a supported combination of versions in the Support Matrix for OpenShift Container Platform on Red Hat Virtualization (RHV).
You are familiar with the OpenShift Container Platform installation and update processes.

10.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster. You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

10.3.3. Requirements for the RHV environment

To install and run an OpenShift Container Platform cluster, the RHV environment must meet the following requirements.

Not meeting these requirements can cause the installation or process to fail. Additionally, not meeting these requirements can cause the OpenShift Container Platform cluster to fail days or weeks after installation.

The following requirements for CPU, memory, and storage resources are based on default values multiplied by the default number of virtual machines the installation program creates. These resources must be available in addition to what the RHV environment uses for non-OpenShift Container Platform operations.

By default, the installation program creates seven virtual machines during the installation process. First, it creates a bootstrap virtual machine to provide temporary services and a control plane while it creates the rest of the OpenShift Container Platform cluster. When the installation program finishes creating the cluster, deleting the bootstrap machine frees up its resources.

If you increase the number of virtual machines in the RHV environment, you must increase the resources accordingly.

**Requirements**

- The RHV environment has one data center whose state is **Up**.
- The RHV data center contains an RHV cluster.
- The RHV cluster has the following resources exclusively for the OpenShift Container Platform cluster:
- Minimum 28 vCPUs: four for each of the seven virtual machines created during installation.

- 112 GiB RAM or more, including:
  - 16 GiB or more for the bootstrap machine, which provides the temporary control plane.
  - 16 GiB or more for each of the three control plane machines which provide the control plane.
  - 16 GiB or more for each of the three compute machines, which run the application workloads.

- The RHV storage domain must meet these etcd backend performance requirements.

- In production environments, each virtual machine must have 120 GiB or more. Therefore, the storage domain must provide 840 GiB or more for the default OpenShift Container Platform cluster. In resource-constrained or non-production environments, each virtual machine must have 32 GiB or more, so the storage domain must have 230 GiB or more for the default OpenShift Container Platform cluster.

- To download images from the Red Hat Ecosystem Catalog during installation and update procedures, the RHV cluster must have access to an internet connection. The Telemetry service also needs an internet connection to simplify the subscription and entitlement process.

- The RHV cluster must have a virtual network with access to the REST API on the RHV Manager. Ensure that DHCP is enabled on this network, because the VMs that the installer creates obtain their IP address by using DHCP.

- A user account and group with the following least privileges for installing and managing an OpenShift Container Platform cluster on the target RHV cluster:
  - DiskOperator
  - DiskCreator
  - UserTemplateBasedVm
  - TemplateOwner
  - TemplateCreator
  - ClusterAdmin on the target cluster

**WARNING**

Apply the principle of least privilege: Avoid using an administrator account with SuperUser privileges on RHV during the installation process. The installation program saves the credentials you provide to a temporary ovirt-config.yaml file that might be compromised.

10.3.4. Verifying the requirements for the RHV environment
Verify that the RHV environment meets the requirements to install and run an OpenShift Container Platform cluster. Not meeting these requirements can cause failures.

**IMPORTANT**

These requirements are based on the default resources the installation program uses to create control plane and compute machines. These resources include vCPUs, memory, and storage. If you change these resources or increase the number of OpenShift Container Platform machines, adjust these requirements accordingly.

**Procedure**

1. Check the RHV version.
   
   a. In the RHV Administration Portal, click the ? help icon in the upper-right corner and select *About*.
   
   b. In the window that opens, make a note of the **RHV Software Version**.
   
   c. Confirm that version 4.6 of OpenShift Container Platform and the version of RHV you noted are one of the supported combinations in the Support Matrix for OpenShift Container Platform on RHV.

2. Inspect the data center, cluster, and storage.
   
   a. In the RHV Administration Portal, click **Compute → Data Centers**.
   
   b. Confirm that the data center where you plan to install OpenShift Container Platform is accessible.
   
   c. Click the name of that data center.
   
   d. In the data center details, on the **Storage** tab, confirm the storage domain where you plan to install OpenShift Container Platform is **Active**.
   
   e. Record the **Domain Name** for use later on.
   
   f. Confirm **Free Space** has at least 230 GiB.
   
   g. Confirm that the storage domain meets these etcd backend performance requirements, which you can measure by using the fio performance benchmarking tool.
   
   h. In the data center details, click the **Clusters** tab.
   
   i. Find the RHV cluster where you plan to install OpenShift Container Platform. Record the cluster name for use later on.

3. Inspect the RHV host resources.
   
   a. In the RHV Administration Portal, click **Compute > Clusters**.
   
   b. Click the cluster where you plan to install OpenShift Container Platform.
   
   c. In the cluster details, click the **Hosts** tab.
   
   d. Inspect the hosts and confirm they have a combined total of at least 28 **Logical CPU Cores** available exclusively for the OpenShift Container Platform cluster.
e. Record the number of available Logical CPU Cores for use later on.

f. Confirm that these CPU cores are distributed so that each of the seven virtual machines created during installation can have four cores.

g. Confirm that, all together, the hosts have 112 GiB of Max free Memory for scheduling new virtual machines distributed to meet the requirements for each of the following OpenShift Container Platform machines:

- 16 GiB required for the bootstrap machine
- 16 GiB required for each of the three control plane machines
- 16 GiB for each of the three compute machines

h. Record the amount of Max free Memory for scheduling new virtual machines for use later on.

4. Verify that the virtual network for installing OpenShift Container Platform has access to the RHV Manager’s REST API. From a virtual machine on this network, use curl to reach the RHV Manager’s REST API:

```bash
$ curl -k -u <username>@<profile>:<password> \ https://<engine-fqdn>/ovirt-engine/api
```

1. For `<username>`, specify the user name of an RHV account with privileges to create and manage an OpenShift Container Platform cluster on RHV. For `<profile>`, specify the login profile, which you can get by going to the RHV Administration Portal login page and reviewing the Profile dropdown list. For `<password>`, specify the password for that user name.

2. For `<engine-fqdn>`, specify the fully qualified domain name of the RHV environment.

For example:

```bash
$ curl -k -u ocpadmin@internal:pw123 \ https://rhv-env.virtlab.example.com/ovirt-engine/api
```

### 10.3.5. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.
You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

**Firewall**

Configure your firewall so your cluster has access to required sites.

See also:

- Red Hat Virtualization Manager firewall requirements
- Host firewall requirements

**Load balancers**

Configure one or preferably two layer-4 load balancers:

- Provide load balancing for ports 6443 and 22623 on the control plane and bootstrap machines. Port 6443 provides access to the Kubernetes API server and must be reachable both internally and externally. Port 22623 must be accessible to nodes within the cluster.

- Provide load balancing for port 443 and 80 for machines that run the Ingress router, which are usually compute nodes in the default configuration. Both ports must be accessible from within and outside the cluster.

**DNS**

Configure infrastructure-provided DNS to allow the correct resolution of the main components and services. If you use only one load balancer, these DNS records can point to the same IP address.

- Create DNS records for `api.<cluster_name>.<base_domain>` (internal and external resolution) and `api-int.<cluster_name>.<base_domain>` (internal resolution) that point to the load balancer for the control plane machines.

- Create a DNS record for `*.apps.<cluster_name>.<base_domain>` that points to the load balancer for the Ingress router. For example, ports 443 and 80 of the compute machines.

**Table 10.6. All machines to all machines**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
</tbody>
</table>
### Network topology requirements

The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

**Load balancers**

Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**

   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

#### Table 10.7. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

#### Table 10.8. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>
### Table 10.10. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>
TIP
If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

NOTE
A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

NTP configuration
OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

10.3.6. Setting up the installation machine
To run the binary openshift-install installation program and Ansible scripts, set up the RHV Manager or a Red Hat Enterprise Linux (RHEL) computer with network access to the RHV environment and the REST API on the Manager.

Procedure

1. Update or install Python3 and Ansible. For example:

   # dnf update python3 ansible

2. Install the python3-ovirt-engine-sdk4 package to get the Python Software Development Kit.

3. Install the ovirt.image-template Ansible role. On the RHV Manager and other Red Hat Enterprise Linux (RHEL) machines, this role is distributed as the ovirt-ansible-image-template package. For example, enter:

   # dnf install ovirt-ansible-image-template

4. Install the ovirt.vminfra Ansible role. On the RHV Manager and other RHEL machines, this role is distributed as the ovirt-ansible-vm-infra package.

   # dnf install ovirt-ansible-vm-infra

5. Create an environment variable and assign an absolute or relative path to it. For example, enter:

   $ export ASSETS_DIR=./wrk
The installation program uses this variable to create a directory where it saves important installation-related files. Later, the installation process reuses this variable to locate those asset files. Avoid deleting this assets directory; it is required for uninstalling the cluster.

10.3.7. Setting up the CA certificate for RHV

Download the CA certificate from the Red Hat Virtualization (RHV) Manager and set it up on the installation machine.

You can download the certificate from a webpage on the RHV Manager or by using a `curl` command. Later, you provide the certificate to the installation program.

Procedure

1. Use either of these two methods to download the CA certificate:
   - Go to the Manager’s webpage, `https://<engine-fqdn>/ovirt-engine/`. Then, under Downloads, click the CA Certificate link.
   - Run the following command:
     ```bash
     ```
     For `<engine-fqdn>`, specify the fully qualified domain name of the RHV Manager, such as `rhv-env.virtlab.example.com`.

2. Configure the CA file to grant rootless user access to the Manager. Set the CA file permissions to have an octal value of `0644` (symbolic value: `-rw-r---`):

   ```bash
   $ sudo chmod 0644 /tmp/ca.pem
   ```

3. For Linux, copy the CA certificate to the directory for server certificates. Use `-p` to preserve the permissions:

   ```bash
   $ sudo cp -p /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
   ```

4. Add the certificate to the certificate manager for your operating system:
   - For macOS, double-click the certificate file and use the Keychain Access utility to add the file to the System keychain.
   - For Linux, update the CA trust:
     ```bash
     $ sudo update-ca-trust
     ```

NOTE

If you use your own certificate authority, make sure the system trusts it.
Additional resources

- To learn more, see Authentication and Security in the RHV documentation.

10.3.8. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N '' \
   -f <path>/<file_name>
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   Agent pid 31874
NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name> 1
```

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

10.3.9. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

IMPORTANT

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.
CHAPTER 10. INSTALLING ON RHV

4. Extract the installation program. For example, on a computer that uses a Linux operating
system, run the following command:
$ tar xvf openshift-install-linux.tar.gz
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager . This pull
secret allows you to authenticate with the services that are provided by the included authorities,
including Quay.io, which serves the container images for OpenShift Container Platform
components.

10.3.10. Downloading the Ansible playbooks
Download the Ansible playbooks for installing OpenShift Container Platform version 4.6 on RHV.
Procedure
On your installation machine, run the following commands:
$ mkdir playbooks
$ cd playbooks
$ curl -s -L -X GET https://api.github.com/repos/openshift/installer/contents/upi/ovirt?
ref=release-4.6 |
grep 'download_url.*\.yml' |
awk '{ print $2 }' | sed -r 's/("|",)//g' |
xargs -n 1 curl -O
Next steps
After you download these Ansible playbooks, you must also create the environment variable for
the assets directory and customize the inventory.yml file before you create an installation
configuration file by running the installation program.

10.3.11. The inventory.yml file
You use the inventory.yml file to define and create elements of the OpenShift Container Platform
cluster you are installing. This includes elements such as the Red Hat Enterprise Linux CoreOS (RHCOS)
image, virtual machine templates, bootstrap machine, control plane nodes, and worker nodes. You also
use inventory.yml to destroy the cluster.
The following inventory.yml example shows you the parameters and their default values. The quantities
and numbers in these default values meet the requirements for running a production OpenShift
Container Platform cluster in a RHV environment.

Example inventory.yml file
--all:
vars:
ovirt_cluster: "Default"
ocp:

1631


assets_dir: "{{ lookup('env', 'ASSETS_DIR') }}"
ovirt_config_path: "{{ lookup('env', 'HOME') }}/.ovirt/ovirt-config.yaml"

# ---
# {op-system} section
# ---
rhcos:
    image_url: "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz"
    local_cmp_image_path: "/tmp/rhcos.qcow2.gz"
    local_image_path: "/tmp/rhcos.qcow2"

# ---
# Profiles section
# ---
control_plane:
    cluster: "{{ ovirt_cluster }}"
    memory: 16GiB
    sockets: 4
    cores: 1
    template: rhcos_tpl
    operating_system: "rhcos_x64"
    type: high_performance
    graphical_console:
        headless_mode: false
        protocol:
            - spice
            - vnc
    disks:
        - size: 120GiB
          name: os
          interface: virtio_scsi
          storage_domain: depot_nvme
    nics:
        - name: nic1
          network: lab
          profile: lab

compute:
    cluster: "{{ ovirt_cluster }}"
    memory: 16GiB
    sockets: 4
    cores: 1
    template: worker_rhcos_tpl
    operating_system: "rhcos_x64"
    type: high_performance
    graphical_console:
        headless_mode: false
        protocol:
            - spice
            - vnc
    disks:
        - size: 120GiB
          name: os
          interface: virtio_scsi
          storage_domain: depot_nvme
IMPORTANT

Enter values for parameters whose descriptions begin with "Enter." Otherwise, you can use the default value or replace it with a new value.

General section

- **ovirt_cluster**: Enter the name of an existing RHV cluster in which to install the OpenShift Container Platform cluster.

- **ocp.assets_dir**: The path of a directory the `openshift-install` installation program creates to store the files that it generates.

- **ocp.ovirt_config_path**: The path of the `ovirt-config.yaml` file the installation program generates, for example, `~/wrk/install-config.yaml`. This file contains the credentials required to interact with the REST API of the Manager.

Red Hat Enterprise Linux CoreOS (RHCOS) section

- **image_url**: Enter the URL of the RHCOS image you specified for download.

- **local_cmp_image_path**: The path of a local download directory for the compressed RHCOS image.
Profiles section

This section consists of two profiles:

- **control_plane**: The profile of the bootstrap and control plane nodes.
- **compute**: The profile of workers nodes in the compute plane.

These profiles have the following parameters. The default values of the parameters meet the minimum requirements for running a production cluster. You can increase or customize these values to meet your workload requirements.

- **cluster**: The value gets the cluster name from `ovirt_cluster` in the General Section.
- **memory**: The amount of memory, in GB, for the virtual machine.
- **sockets**: The number of sockets for the virtual machine.
- **cores**: The number of cores for the virtual machine.
- **template**: The name of the virtual machine template. If plan to install multiple clusters, and these clusters use templates that contain different specifications, prepend the template name with the ID of the cluster.
- **operating_system**: The type of guest operating system in the virtual machine. With oVirt/RHV version 4.4, this value must be `rhcos_x64` so the value of Ignition script can be passed to the VM.
- **type**: Enter `server` as the type of the virtual machine.

**IMPORTANT**

You must change the value of the type parameter from `high_performance` to `server`.

- **disks**: The disk specifications. The **control_plane** and **compute** nodes can have different storage domains.
- **size**: The minimum disk size.
- **name**: Enter the name of a disk connected to the target cluster in RHV.
- **interface**: Enter the interface type of the disk you specified.
- **storage_domain**: Enter the storage domain of the disk you specified.
- **nics**: Enter the **name** and **network** the virtual machines use. You can also specify the virtual network interface profile. By default, NICs obtain their MAC addresses from the oVirt/RHV MAC pool.

Virtual machines section

This final section, `vms`, defines the virtual machines you plan to create and deploy in the cluster. By default, it provides the minimum number of control plane and worker nodes for a production environment.
vms contains three required elements:

- **name**: The name of the virtual machine. In this case, `metadata.infraID` prepends the virtual machine name with the infrastructure ID from the `metadata.yml` file.

- **ocp_type**: The role of the virtual machine in the OCP cluster. Possible values are `bootstrap`, `master`, `worker`.

- **profile**: The name of the profile from which each virtual machine inherits specifications. Possible values in this example are `control_plane` or `compute`. You can override the value a virtual machine inherits from its profile. To do this, you add the name of the profile attribute to the virtual machine in `inventory.yml` and assign it an overriding value. To see an example of this, examine the name: `{{ metadata.infraID }}-bootstrap` virtual machine in the preceding `inventory.yml` example: It has a `type` attribute whose value, `server`, overrides the value of the `type` attribute this virtual machine would otherwise inherit from the `control_plane` profile.

**Metadata variables**

For virtual machines, `metadata.infraID` prepends the name of the virtual machine with the infrastructure ID from the `metadata.json` file you create when you build the Ignition files.

The playbooks use the following code to read `infraID` from the specific file located in the `ocp.assets_dir`.

```yaml
---
- name: include metadata.json vars
  include_vars:
    file: "{{ ocp.assets_dir }}/metadata.json"
    name: metadata
...```

10.3.12. Specifying the RHCOS image settings

Update the Red Hat Enterprise Linux CoreOS (RHCOS) image settings of the `inventory.yml` file. Later, when you run this file one of the playbooks, it downloads a compressed Red Hat Enterprise Linux CoreOS (RHCOS) image from the `image_url` URL to the `local_cmp_image_path` directory. The playbook then uncompresses the image to the `local_image_path` directory and uses it to create oVirt/RHV templates.

**Procedure**

1. Locate the RHCOS image download page for the version of OpenShift Container Platform you are installing, such as `Index of /pub/openshift-v4/dependencies/rhcos/latest/latest`.

2. From that download page, copy the URL of an OpenStack `qcow2` image, such as `https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz`.

3. Edit the `inventory.yml` playbook you downloaded earlier. In it, paste the URL as the value for `image_url`. For example:

   ```yaml
   rhcos:
      "https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/4.6/latest/rhcos-openstack.x86_64.qcow2.gz"
   ```
10.3.13. Creating the install config file

You create an installation configuration file by running the installation program, `openshift-install`, and responding to its prompts with information you specified or gathered earlier.

When you finish responding to the prompts, the installation program creates an initial version of the `install-config.yaml` file in the assets directory you specified earlier, for example, `/wrk/install-config.yaml`.

The installation program also creates a file, `$HOME/.ovirt/ovirt-config.yaml`, that contains all the connection parameters that are required to reach the Manager and use its REST API.

**NOTE:** The installation process does not use values you supply for some parameters, such as `Internal API virtual IP` and `Ingress virtual IP`, because you have already configured them in your infrastructure DNS.

It also uses the values you supply for parameters in `inventory.yml`, like the ones for `oVirt cluster`, `oVirt storage`, and `oVirt network`. And uses a script to remove or replace these same values from `install-config.yaml` with the previously mentioned `virtual IPs`.

**Procedure**

1. Run the installation program:

   ```
   $ openshift-install create install-config --dir $ASSETS_DIR
   ```

2. Respond to the installation program’s prompts with information about your system.

   **Example output**

   ```
   ? SSH Public Key /home/user/.ssh/id_dsa.pub
   ? Platform <ovirt>
   ? Enter ovirt-engine username <ocpadmin@internal>
   ? Enter password <******>
   ? oVirt cluster <cluster>
   ? oVirt storage <storage>
   ? oVirt network <net>
   ? Internal API virtual IP <172.16.0.252>
   ? Ingress virtual IP <172.16.0.251>
   ? Base Domain <example.org>
   ? Cluster Name <ocp4>
   ? Pull Secret [? for help] <*******>
   ```

   For `Internal API virtual IP` and `Ingress virtual IP`, supply the IP addresses you specified when you configured the DNS service.

   Together, the values you enter for the `oVirt cluster` and `Base Domain` prompts form the FQDN portion of URLs for the REST API and any applications you create, such as `https://api.ocp4.example.org:6443/` and `https://console-openshift-console.apps.ocp4.example.org`.

   You can get the pull secret from the Red Hat OpenShift Cluster Manager.
10.3.14. Customizing install-config.yaml

Here, you use three Python scripts to override some of the installation program’s default behaviors:

- By default, the installation program uses the machine API to create nodes. To override this default behavior, you set the number of compute nodes to zero replicas. Later, you use Ansible playbooks to create the compute nodes.

- By default, the installation program sets the IP range of the machine network for nodes. To override this default behavior, you set the IP range to match your infrastructure.

- By default, the installation program sets the platform to ovirt. However, installing a cluster on user-provisioned infrastructure is more similar to installing a cluster on bare metal. Therefore, you delete the ovirt platform section from install-config.yaml and change the platform to none. Instead, you use inventory.yml to specify all of the required settings.

NOTE

These snippets work with Python 3 and Python 2.

Procedure

1. Set the number of compute nodes to zero replicas:

   ```
   $ python3 -c 'import os, yaml
   path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
   conf = yaml.safe_load(open(path))
   conf["compute"][0]["replicas"] = 0
   open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
   ```

2. Set the IP range of the machine network. For example, to set the range to 172.16.0.0/16, enter:

   ```
   $ python3 -c 'import os, yaml
   path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
   conf = yaml.safe_load(open(path))
   conf["networking"]["machineNetwork"][0]["cidr"] = "172.16.0.0/16"
   open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
   ```

3. Remove the ovirt section and change the platform to none:

   ```
   $ python3 -c 'import os, yaml
   path = "%s/install-config.yaml" % os.environ["ASSETS_DIR"]
   conf = yaml.safe_load(open(path))
   platform = conf["platform"]
   del platform["ovirt"]
   platform["none"] = {}
   open(path, "w").write(yaml.dump(conf, default_flow_style=False))'
   ```

10.3.15. Generate manifest files

Use the installation program to generate a set of manifest files in the assets directory.

The command to generate the manifest files displays a warning message before it consumes the install-config.yaml file.
If you plan to reuse the `install-config.yaml` file, create a backup copy of it before you back it up before you generate the manifest files.

**Procedure**

1. Optional: Create a backup copy of the `install-config.yaml` file:

   ```bash
   $ cp install-config.yaml install-config.yaml.backup
   
   2. Generate a set of manifests in your assets directory:

   ```bash
   $ openshift-install create manifests --dir $ASSETS_DIR
   
   This command displays the following messages.

   **Example output**

   ```
   INFO Consuming Install Config from target directory
   WARNING Making control-plane schedulable by setting MastersSchedulable to true for
   Scheduler cluster settings
   
   The command generates the following manifest files:

   **Example output**

   ```bash
   $ tree
   .
   ├── wrk
   │   └── manifests
   │       ├── 04-open-shift-machine-config-operator.yaml
   │       ├── cluster-config.yaml
   │       ├── cluster-dns-02-config.yml
   │       ├── cluster-infrastructure-02-config.yml
   │       ├── cluster-ingress-02-config.yml
   │       ├── cluster-network-01-crd.yml
   │       ├── cluster-network-02-config.yml
   │       ├── cluster-proxy-01-config.yaml
   │       ├── cluster-scheduler-02-config.yml
   │       ├── cvo-overrides.yaml
   │       ├── etcd-ca-bundle-configmap.yaml
   │       ├── etcd-client-secret.yaml
   │       ├── etcd-host-service-endpoints.yaml
   │       ├── etcd-host-service.yaml
   │       ├── etcd-metric-client-secret.yaml
   │       ├── etcd-metric-serving-ca-configmap.yaml
   │       └── etcd-namespace.yaml
   │                   ├── etcd-service.yaml
   │                   └── openshift-config-secret-pull-secret.yaml
   ```
Next steps

- Make control plane nodes non-schedulable.

10.3.16. Making control-plane nodes non-schedulable

Because you are manually creating and deploying the control plane machines, you must configure a manifest file to make the control plane nodes non-schedulable.

Procedure

1. To make the control plane nodes non-schedulable, enter:

   ```
   $ python3 -c 'import os, yaml
   path = "%s/manifests/cluster-scheduler-02-config.yml" % os.environ["ASSETS_DIR"]
   data = yaml.safe_load(open(path))
   data["spec"]['mastersSchedulable'] = False
   open(path, "w").write(yaml.dump(data, default_flow_style=False))'
   ```

10.3.17. Building the Ignition files

To build the Ignition files from the manifest files you just generated and modified, you run the installation program. This action creates a Red Hat Enterprise Linux CoreOS (RHCOS) machine, initramfs, which fetches the Ignition files and performs the configurations needed to create a node.

In addition to the Ignition files, the installation program generates the following:

- An auth directory that contains the admin credentials for connecting to the cluster with the oc and kubectl utilities.

- A metadata.json file that contains information such as the OpenShift Container Platform cluster name, cluster ID, and infrastructure ID for the current installation.

The Ansible playbooks for this installation process use the value of infraID as a prefix for the virtual machines they create. This prevents naming conflicts when there are multiple installations in the same oVirt/RHV cluster.

**NOTE**

Certificates in Ignition configuration files expire after 24 hours. Complete the cluster installation and keep the cluster running in a non-degraded state for 24 hours so that the first certificate rotation can finish.

Procedure

1. To build the Ignition files, enter:
10.3.18. Creating templates and virtual machines

After confirming the variables in the `inventory.yml`, you run the first Ansible provisioning playbook, `create-templates-and-vms.yml`.

This playbook uses the connection parameters for the RHV Manager from `$HOME/.ovirt/ovirt-config.yaml` and reads `metadata.json` in the assets directory.

If a local Red Hat Enterprise Linux CoreOS (RHCOS) image is not already present, the playbook downloads one from the URL you specified for `image_url` in `inventory.yml`. It extracts the image and uploads it to RHV to create templates.

The playbook creates a template based on the `control_plane` and `compute` profiles in the `inventory.yml` file. If these profiles have different names, it creates two templates.

When the playbook finishes, the virtual machines it creates are stopped. You can get information from them to help configure other infrastructure elements. For example, you can get the virtual machines’ MAC addresses to configure DHCP to assign permanent IP addresses to the virtual machines.

Procedure

1. In `inventory.yml`, under the `control_plane` and `compute` variables, change both instances of `type: high_performance` to `type: server`.

2. Optional: If you plan to perform multiple installations to the same cluster, create different templates for each OCP installation. In the `inventory.yml` file, prepend the value of `template` with `infraID`. For example:

   ```yaml
   control_plane:
      cluster: "{{ ovirt_cluster }}"
      memory: 16GiB
      sockets: 4
      cores: 1
      template: "{{ metadata.infraID }}-rhcos_tpl"
      operating_system: "rhcos_x64"
      ...
   ```

3. Create the templates and virtual machines:

   ```bash
   $ openshift-install create ignition-configs --dir $ASSETS_DIR
   ```

   Example output

   ```bash
   $ tree
   .
   ├── wrk
   │   └── auth
   │       ├── kubeadmin-password
   │       └── kubeconfig
   │          └── bootstrap.ign
   │                 └── master.ign
   │                        └── metadata.json
   │                                           └── worker.ign
   ```
10.3.19. Creating the bootstrap machine

You create a bootstrap machine by running the bootstrap.yml playbook. This playbook starts the bootstrap virtual machine, and passes it the bootstrap.ign Ignition file from the assets directory. The bootstrap node configures itself so it can serve Ignition files to the control plane nodes.

To monitor the bootstrap process, you use the console in the RHV Administration Portal or connect to the virtual machine by using SSH.

Procedure

1. Create the bootstrap machine:

   $ ansible-playbook -i inventory.yml bootstrap.yml

2. Connect to the bootstrap machine using a console in the Administration Portal or SSH. Replace <bootstrap_ip> with the bootstrap node IP address. To use SSH, enter:

   $ ssh core@<bootstrap.ip>

3. Collect bootkube.service journal unit logs for the release image service from the bootstrap node:

   [core@ocp4-lk6b4-bootstrap ~]$ journalctl -b -f -u release-image.service -u bootkube.service

   **NOTE**

   The bootkube.service log on the bootstrap node outputs etcd connection refused errors, indicating that the bootstrap server is unable to connect to etcd on control plane nodes (also known as the master nodes). After etcd has started on each control plane node and the nodes have joined the cluster, the errors should stop.

10.3.20. Creating the control plane nodes

You create the control plane nodes by running the masters.yml playbook. This playbook passes the master.ign Ignition file to each of the virtual machines. The Ignition file contains a directive for the control plane node to get the Ignition from a URL such as https://api-int.ocp4.example.org:22623/config/master. The port number in this URL is managed by the load balancer, and is accessible only inside the cluster.

Procedure

1. Create the control plane nodes:

   $ ansible-playbook -i inventory.yml masters.yml

2. While the playbook creates your control plane, monitor the bootstrapping process:

   $ openshift-install wait-for bootstrap-complete --dir $ASSETS_DIR
Example output

INFO API v1.18.3+b74c5ed up
INFO Waiting up to 40m0s for bootstrapping to complete...

3. When all the pods on the control plane nodes and etcd are up and running, the installation program displays the following output.

Example output

INFO It is now safe to remove the bootstrap resources

10.3.21. Verifying cluster status

You can verify your OpenShift Container Platform cluster’s status during or after installation.

Procedure

1. In the cluster environment, export the administrator’s kubeconfig file:

   $ export KUBECONFIG=$ASSETS_DIR/auth/kubeconfig

   The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server.

2. View the control plane and compute machines created after a deployment:

   $ oc get nodes

3. View your cluster’s version:

   $ oc get clusterversion

4. View your Operators’ status:

   $ oc get clusteroperator

5. View all running pods in the cluster:

   $ oc get pods -A

10.3.22. Removing the bootstrap machine

After the wait-for command shows that the bootstrap process is complete, you must remove the bootstrap virtual machine to free up compute, memory, and storage resources. Also, remove settings for the bootstrap machine from the load balancer directives.

Procedure

1. To remove the bootstrap machine from the cluster, enter:

   $ ansible-playbook -i inventory.yml retire-bootstrap.yml
2. Remove settings for the bootstrap machine from the load balancer directives.

10.3.23. Creating the worker nodes and completing the installation

Creating worker nodes is similar to creating control plane nodes. However, worker nodes workers do not automatically join the cluster. To add them to the cluster, you review and approve the workers’ pending CSRs (Certificate Signing Requests).

After approving the first requests, you continue approving CSR until all of the worker nodes are approved. When you complete this process, the worker nodes become **Ready** and can have pods scheduled to run on them.

Finally, monitor the command line to see when the installation process completes.

Procedure

1. Create the worker nodes:

   $ ansible-playbook -i inventory.yml workers.yml

2. To list all of the CSRs, enter:

   $ oc get csr -A

   Eventually, this command displays one CSR per node. For example:

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>SIGNERNAME</th>
<th>REQUESTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-2lnxd</td>
<td>63m</td>
<td>kubernetes.io/kubelet-serving</td>
<td>system:node:ocp4-lk6b4-master0.ocp4.example.org</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-hff4q</td>
<td>64m</td>
<td>kubernetes.io/kube-apiserver-client-kubelet</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-hsn96</td>
<td>60m</td>
<td>kubernetes.io/kubelet-serving</td>
<td>system:node:ocp4-lk6b4-master2.ocp4.example.org</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-m724n</td>
<td>6m2s</td>
<td>kubernetes.io/kube-apiserver-client-kubelet</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pending</td>
</tr>
<tr>
<td>csr-p4dz2</td>
<td>60m</td>
<td>kubernetes.io/kube-apiserver-client-kubelet</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-t9vfj</td>
<td>60m</td>
<td>kubernetes.io/kubelet-serving</td>
<td>system:node:ocp4-lk6b4-master1.ocp4.example.org</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-tgtr</td>
<td>61m</td>
<td>kubernetes.io/kube-apiserver-client-kubelet</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Approved,Issued</td>
</tr>
<tr>
<td>csr-wcbrf</td>
<td>7m6s</td>
<td>kubernetes.io/kube-apiserver-client-kubelet</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pending</td>
</tr>
</tbody>
</table>

3. To filter the list and see only pending CSRs, enter:

   $ watch "oc get csr -A | grep pending -i"
This command refreshes the output every two seconds and displays only pending CSRs. For example:

### Example output

```bash
every 2.0s: oc get csr -A | grep pending -i
csr-m724n   10m   kubernetes.io/kube-apiserver-client-kubelet
            system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-wcbrf   11m   kubernetes.io/kube-apiserver-client-kubelet
            system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
```

4. Inspect each pending request. For example:

### Example output

```
$ oc describe csr csr-m724n
```

### Example output

```
Name:               csr-m724n
Labels:             <none>
Annotations:        <none>
CreationTimestamp:  Sun, 19 Jul 2020 15:59:37 +0200
Signer:             kubernetes.io/kube-apiserver-client-kubelet
Status:             Pending
Subject:
    Common Name:    system:node:ocp4-lk6b4-worker1.ocp4.example.org
    Serial Number:  <none>
    Organization:   system:nodes
Events: <none>
```

5. If the CSR information is correct, approve the request:

```
$ oc adm certificate approve csr-m724n
```

6. Wait for the installation process to finish:

```
$ openshift-install wait-for install-complete --dir $ASSETS_DIR --log-level debug
```

When the installation completes, the command line displays the URL of the OpenShift Container Platform web console and the administrator user name and password.

### 10.3.24. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service.

10.4. UNINSTALLING A CLUSTER ON RHV

You can remove an OpenShift Container Platform cluster from Red Hat Virtualization (RHV).

10.4.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

- Have a copy of the installation program that you used to deploy the cluster.
- Have the files that the installation program generated when you created your cluster.

Procedure

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

   ```bash
   $ ./openshift-install destroy cluster \
   --dir <installation_directory> --log-level info
   ```

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   2 To view different details, specify warn, debug, or error instead of info.

   NOTE

   You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the metadata.json file in this directory to delete the cluster.

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.

10.4.2. Removing a cluster that uses user-provisioned infrastructure
When you are finished using the cluster, you can remove a cluster that uses user-provisioned infrastructure from your cloud.

**Prerequisites**

- Have the original playbook files, assets directory and files, and `$ASSETS_DIR` environment variable that you used to you install the cluster. Typically, you can achieve this by using the same computer you used when you installed the cluster.

**Procedure**

1. To remove the cluster, enter:

   ```
   $ ansible-playbook -i inventory.yml \
   retire-bootstrap.yml \
   retire-masters.yml \
   retire-workers.yml
   ```

2. Remove any configurations you added to DNS, load balancers, and any other infrastructure for this cluster.
CHAPTER 11. INSTALLING A CLUSTER ON VSPHERE

11.1. INSTALLING A CLUSTER ON VSPHERE

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure.

11.1.1. Prerequisites

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.
- Review details about the OpenShift Container Platform installation and update processes.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

11.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

11.1.3. VMware vSphere infrastructure requirements
You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

Table 11.1. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

11.1.4. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

**Required vCenter account privileges**

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.
An additional role is required if the installation program is to create a vSphere virtual machine folder.

**Example 11.1. Roles and privileges required for installation**

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>Cns.Searchable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.AttachTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sessions.ValidateSession</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StorageProfile.View</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>Host.Config.Storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.AssignResourcePool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 11.2. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td>Installation program creates the folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see *vSphere Permissions and User Management Tasks* in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.

Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.
Cluster resources
When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits
Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements
You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses
An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

### Table 11.2. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;</code></td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td><code>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;</code></td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

### 11.1.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

#### Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:
Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```bash
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   ```bash
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

   **Next steps**

   - When you install OpenShift Container Platform, provide the SSH public key to the installation program.

   **11.1.6. Obtaining the installation program**

   Before you install OpenShift Container Platform, download the installation file on a local computer.

   **Prerequisites**
You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.1.7. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.

2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

   certs
     └── lin
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

### 11.1.8. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \[1
--log-level=info \[2
```

[1] `<installation_directory>`: The directory containing the installation program.

[2] `-log-level=info`: Set the log level to information. Change the level according to your preference.
1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Provide values at the prompts:

a. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

b. Select `vsphere` as the platform to target.

c. Specify the name of your vCenter instance.

d. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
   The installation program connects to your vCenter instance.

e. Select the datacenter in your vCenter instance to connect to.

f. Select the default vCenter datastore to use.

h. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

i. Enter the virtual IP address that you configured for control plane API access.

j. Enter the virtual IP address that you configured for cluster ingress.

k. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

l. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

m. Paste the pull secret from the Red Hat OpenShift Cluster Manager.
When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the **kubeadmin** user, display in your terminal.

**Example output**

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

### 11.1.9. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) in order to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

11.1.9.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

11.1.9.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.

   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
11.1.9.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ``

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

11.1.10. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```
11.1.11. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

11.1.11.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

11.1.11.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

11.1.11.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.
IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is **emptyDir**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```yaml
   storage:
   pvc:
     claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.
4. Check the `clusteroperator` status:

```bash
$ oc get clusteroperator image-registry
```

11.11.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only 1 replica:

```bash
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate","replicas":1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere `PersistentVolumeClaim` object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   1. A unique name that represents the `PersistentVolumeClaim` object.
   2. The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.
   3. The access mode of the persistent volume claim. With `ReadWriteOnce`, the volume can be mounted with read and write permissions by a single node.
   4. The size of the persistent volume claim.

   b. Create the `PersistentVolumeClaim` object from the file:
$ oc create -f pvc.yaml -n openshift-image-registry

3. Edit the registry configuration so that it references the correct PVC:

$ oc edit config.imageregistry.operator.openshift.io -o yaml

Example output

```
storage:
pvc:
  claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see Configuring the registry for vSphere.

11.1.12. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.

2. Clone the persistent volume.

3. Restart the application.

4. Create a backup of the cloned volume.

5. Delete the cloned volume.

11.1.13. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service
11.1.14. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

11.2. INSTALLING A CLUSTER ON VSPHERE WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

11.2.1. Prerequisites

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide `ReadWriteMany` access modes.
- Review details about the OpenShift Container Platform installation and update processes.
- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

11.2.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
11.2.3. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

Table 11.3. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

11.2.4. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.
**Required vCenter account privileges**

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>Cns.Searchable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.AttachTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sessions.ValidateSession</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StorageProfile.View</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>Host.Config.Storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.AssignResourcePool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 11.4. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td>Installation program creates the folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see *vSphere Permissions and User Management Tasks* in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute–only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.

Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.
Cluster resources
When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits
Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements
You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

NOTE
It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses
An installer-provisioned vSphere installation requires two static IP addresses:

- The API address is used to access the cluster API.
- The Ingress address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`. 

### Table 11.4. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

#### 11.2.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:
Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the **ssh-agent** process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   
   **Example output**
   
   Agent pid 31874
   
   **NOTE**
   
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.
   
3. Add your SSH private key to the **ssh-agent**:  

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**11.2.6. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**
• You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.2.7. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The <vCenter>/certs/download.zip file downloads.

2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

   certs
   └── lin
3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

### 11.2.8. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

#### Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

#### Procedure

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>  ❶
```

   ❶ For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select vsphere as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster. The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the **Installation configuration parameters** section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
IMPORTANT

The install-config.yaml file is consumed during the installation process. If you want to reuse the file, you must back it up now.

11.2.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the install-config.yaml installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the install-config.yaml file to provide more details about the platform.

NOTE

After installation, you cannot modify these parameters in the install-config.yaml file.

IMPORTANT

The openshift-install command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

11.2.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 11.5. Required parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the install-config.yaml content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;.&lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
| pullSecret   | Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io. | \{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
\}|

11.2.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 11.6. Network parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td>You cannot modify parameters specified by the <code>networking</code> object after installation.</td>
</tr>
<tr>
<td></td>
<td>networking.network Type</td>
<td>Either <code>OpenShiftSDN</code> or <code>OVNKubernetes</code>. The default value is <code>OpenShiftSDN</code>.</td>
</tr>
<tr>
<td></td>
<td>networking.clusterNetwork</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
</tr>
<tr>
<td></td>
<td>networking.clusterNetwork.cidr</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td></td>
<td>networking.clusterNetwork.hostPrefix</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>networking.serviceNetwork</td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
</tr>
<tr>
<td></td>
<td>networking.serviceNetwork.cidr</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td></td>
<td>networking.machineNetwork</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>

1680
networking.machineNetwork.cidr

Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.

An IP network block in CIDR notation. For example, 10.0.0.0/16.

**NOTE**

Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

### 11.2.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 11.7. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of MachinePool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hypervethreading</em>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
Enable or disable FIPS mode. The default is `false` (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fips</code></td>
<td>Sources and repositories for the release-image content.</td>
<td><code>false</code> or <code>true</code></td>
</tr>
<tr>
<td><code>imageContentSources</code></td>
<td>Required if you use <code>imageContentSources</code>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>Array of objects. Includes a <code>source</code> and, optionally, <code>mirrors</code>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><code>imageContentSources.source</code></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>String</td>
</tr>
</tbody>
</table>

```
Parameter | Description                          | Values   |
----------|--------------------------------------|----------|
| fips     | Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead. | false or true |
```
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**publish**

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>publish</td>
<td>How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.</td>
<td>Internal or External. The default value is External. Setting this field to Internal is not supported on non-cloud platforms.</td>
</tr>
</tbody>
</table>

**sshKey**

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

| sshKey    | The SSH key or keys to authenticate access your cluster machines. | One or more keys. For example: sshKey: <key1> <key2> <key3> |

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

### 11.2.8.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table:

**Table 11.8. Additional VMware vSphere cluster parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.vCenter</td>
<td>The fully-qualified hostname or IP address of the vCenter server.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.username</td>
<td>The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform.vsphere.password</strong></td>
<td>The password for the vCenter user name.</td>
<td>String</td>
</tr>
<tr>
<td><strong>platform.vsphere.datacenter</strong></td>
<td>The name of the datacenter to use in the vCenter instance.</td>
<td>String</td>
</tr>
<tr>
<td><strong>platform.vsphere.defaultDatastore</strong></td>
<td>The name of the default datastore to use for provisioning volumes.</td>
<td>String</td>
</tr>
<tr>
<td><strong>platform.vsphere.folder</strong></td>
<td>Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the datacenter virtual machine folder.</td>
<td>String, for example, <code>/&lt;datacenter_name&gt;/vm/&lt;folder_name&gt;/&lt;subfolder_name&gt;</code>.</td>
</tr>
<tr>
<td><strong>platform.vsphere.network</strong></td>
<td>The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.</td>
<td>String</td>
</tr>
<tr>
<td><strong>platform.vsphere.cluster</strong></td>
<td>The vCenter cluster to install the OpenShift Container Platform cluster in.</td>
<td>String</td>
</tr>
<tr>
<td><strong>platform.vsphere.apiVIP</strong></td>
<td>The virtual IP (VIP) address that you configured for control plane API access.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><strong>platform.vsphere.ingressVIP</strong></td>
<td>The virtual IP (VIP) address that you configured for cluster ingress.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
</tbody>
</table>

### 11.2.8.15. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform.vsphere.clusterOSImage</strong></td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <a href="https://mirror.openshift.com/images/rhcos-">https://mirror.openshift.com/images/rhcos-</a>&lt;version&gt;-vmware.&lt;architecture&gt;.ova.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>platform.vsphere.osDisk.diskSizeGB</td>
<td>The size of the disk in gigabytes.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere cpus</td>
<td>The total number of virtual processor cores to assign a virtual machine.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.coresPerSocket</td>
<td>The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket. The default value is 1</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.memoryMB</td>
<td>The size of a virtual machine’s memory in megabytes.</td>
<td>Integer</td>
</tr>
</tbody>
</table>

### 11.2.8.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 3
    platform:
      vsphere:
        cpus: 2
        coresPerSocket: 2
        memoryMB: 8196
        osDisk:
          diskSizeGB: 120
      controlPlane:
        hyperthreading: Enabled
        name: master
        replicas: 3
        platform:
          vsphere:
            cpus: 4
            coresPerSocket: 2
            memoryMB: 16384
            osDisk:
              diskSizeGB: 120
      metadata:
        name: cluster
        platform:
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, `-`, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines` cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.

The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

### 11.2.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
  httpsProxy: https://<username>:<pswd>@<ip>:<port>
  noProxy: example.com
additionalTrustBundle:
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   2. A proxy URL to use for creating HTTPS connections outside the cluster.

   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

   4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 11.2.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

#### Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

#### Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \ 1
   --log-level=info 2
   For `<installation_directory>`, specify the location of your customized `/install-config.yaml` file.
   To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.
   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

   ... INFO Install complete! INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig' INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL" INFO Time elapsed: 36m22s
   ```
NOTE
The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.

IMPORTANT
The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT
You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

11.2.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.

IMPORTANT
If you installed an earlier version of `oc`, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of `oc`.

11.2.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xzvf <file>

5. Place the oc binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

11.2.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.

   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:

   C:\> oc <command>

11.2.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

   $ tar xvzf <file>

   $ echo $PATH

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:

   C:\> oc <command>
5. Move the `oc` binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 11.2.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

   Example output

```
system:admin
```

### 11.2.12. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

### 11.2.12.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from **Removed** to **Managed**.
NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

11.2.12.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

11.2.12.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

IMPORTANT

OpenShift Container Platform supports ReadWriteOnce access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, ReadWriteMany access is required.

- Must have "100Gi" capacity.

IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.
Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   
   **NOTE**
   
   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   
   **Example output**
   
   ```storage:
   pvc:
   claim: 1`
   
   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```
   $ oc get clusteroperator image-registry

11.2.12.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only 1 replica:
2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a **pvc.yaml** file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   
   1 A unique name that represents the **PersistentVolumeClaim** object.
   2 The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
   3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
   4 The size of the persistent volume claim.

   b. Create the **PersistentVolumeClaim** object from the file:

   ```bash
   $ oc create -f pvc.yaml -n openshift-image-registry
   
   $ oc edit config.imageregistry.operator.openshift.io -o yaml
   
   Example output
   
   storage:
   pvc:
     claim: 1
   
   1 Creating a custom PVC allows you to leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).
11.2.13. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

11.2.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

11.2.15. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

11.3. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance by using installer-provisioned infrastructure with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations. To customize the installation, you modify parameters in the install-config.yaml file before you install the cluster.

You must set most of the network configuration parameters during installation, and you can modify only kubeProxy configuration parameters in a running cluster.
11.3.1. Prerequisites

- Provision **persistent storage** for your cluster. To deploy a private image registry, your storage must provide **ReadWriteMany** access modes.

- Review details about the **OpenShift Container Platform installation and update** processes.

- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.

- If you use a firewall, confirm with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.

- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

11.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access **Quay.io** to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

11.3.3. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 11.10. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Component | Minimum supported versions | Description
--- | --- | ---
Hypervisor | vSphere 6.5 and later with HW version 13 | This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the [Red Hat Enterprise Linux 8 supported hypervisors list](#).

Storage with in-tree drivers | vSphere 6.5 and later | This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.

Optional: Networking (NSX-T) | vSphere 6.5U3 or vSphere 6.7U2 and later | vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.

---

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](#) in the VMware documentation.

11.3.4. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

**Required vCenter account privileges**

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

**Example 11.5. Roles and privileges required for installation**
<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
</table>
| vSphere vCenter                  | Always        | Cns.Searchable
InventoryService.Tagging.AttachTag
InventoryService.Tagging.CreateCategory
InventoryService.Tagging.CreateTag
InventoryService.Tagging.DeleteCategory
InventoryService.Tagging.DeleteTag
InventoryService.Tagging.EditCategory
InventoryService.Tagging.EditTag
Sessions.ValidateSession
StorageProfile.View               |
| vSphere vCenter Cluster          | Always        | Host.Config.Storage
Resource.AssignVMToPool
VApp.AssignResourcePool
VApp.Import
VirtualMachine.Config.AddNewDisk |
| vSphere Datastore                | Always        | Datastore.AllocateSpace
Datastore.Browse
Datastore.FileManagement          |
<p>| vSphere Port Group               | Always        | Network.Assign                                                                     |</p>
<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddExistingDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddRemoveDevice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AdvancedConfig</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Annotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.CPUCount</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.DiskExtend</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.DiskLease</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.EditDevice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Memory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RemoveDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Rename</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RestGuestInfo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Rest</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Create</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.CreateFromExisting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Provisioning.Clone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.GuestControl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOff</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.Rest</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Settings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UnknownHardwareUpdate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UnknownHardwareRemove</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UnknownHardwarEUpgrade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UnknownHardwarERemove</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>If the installation program creates the virtual machine folder</td>
<td>Resource.AssignVMTToPool VApp.Import VirtualMachine.Config.Add ExistingDisk NewDisk</td>
</tr>
</tbody>
</table>

OpenShift Container Platform 4.6 Installing 1702
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 11.6. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Installation program creates the folder</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see *vSphere Permissions and User Management Tasks* in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.
Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

**Cluster resources**

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

**Cluster limits**

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

**Networking requirements**

You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

**Required IP Addresses**

An installer-provisioned vSphere installation requires two static IP addresses:

- The API address is used to access the cluster API.
- The Ingress address is used for cluster ingress traffic.
You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 11.11. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;</code></td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td><code>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;</code></td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

11.3.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**
In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure
1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" \\n-f <path>/file_name>
```

Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```
$ eval "$_ssh-agent -s"
```

**Example output**

Agent pid 31874

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/file_name>
```

**Example output**

Identity added: /home/<you>/<path>/file_name> (<computer_name>)

Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**11.3.6. Obtaining the installation program**
Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.3.7. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   └── 7e757f6a.0
│       ├── 8e4f8471.0
│       └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   └── 7e757f6a.0
│       ├── 8e4f8471.0
│       └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crt
    ├── 7e757f6a.0.crt
    └── 8e4f8471.0.crt
```
3 directories, 15 files

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

### 11.3.8. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>
```

1708
1 For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `vsphere` as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.

The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.
2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 11.3.8.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 11.3.8.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter | Description | Values
--- | --- | ---
**baseDomain** | The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the `baseDomain` and `metadata.name` parameter values that uses the `<metadata.name>.<baseDomain>` format. | A fully-qualified domain or subdomain name, such as example.com.

**metadata** | Kubernetes resource `ObjectMeta`, from which only the `name` parameter is consumed. | Object

**metadata.name** | The name of the cluster. DNS records for the cluster are all subdomains of `{{.metadata.name}}.{{.baseDomain}}`. | String of lowercase letters, hyphens (-), and periods (.), such as dev.

**platform** | The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about `platform.<platform>` parameters, consult the following table for your specific platform. | Object
Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.

```
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
```

### 11.3.8.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 11.13. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>networking</strong></td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>networking.network</strong></td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td><strong>networking.clusterNetwork</strong></td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>
|                      | The default value is 10.128.0.0/14 with a host prefix of /23.               | networking:
|                      | If you specify multiple IP address blocks, the blocks must not overlap.     | clusterNetwork:
|                      |                                                                           | - cidr: 10.128.0.0/14 hostPrefix: 23 |

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
### Table 11.14. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td></td>
<td>An IPv4 network.</td>
<td></td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides $2^{(32 - 23)} - 2$ pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16.</td>
<td>An array with an IP address block in CIDR format. For example: networking: serviceNetwork: - 172.30.0.0/16</td>
</tr>
<tr>
<td></td>
<td>The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td></td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines.</td>
<td>An array of objects. For example: networking: machineNetwork: - cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td></td>
<td>If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td></td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
<tr>
<td></td>
<td>The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>NOTE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.</td>
</tr>
</tbody>
</table>

### 11.3.8.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 11.14. Optional parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are amd64 (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td></td>
<td>IMPORTANT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use compute. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use compute. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the controlPlane.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or{}</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <strong>MachinePool</strong> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool.</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Currently, heterogeneous clusters are not supported, so all pools must</td>
<td></td>
</tr>
<tr>
<td></td>
<td>specify the same architecture. Valid values are <strong>amd64</strong> (the default).</td>
<td></td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hypertreading</strong>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td><strong>Enabled</strong> or <strong>Disabled</strong></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td><strong>master</strong></td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud</td>
<td><strong>aws, azure, gcp, openstack, ovirt, vsphere, or{}</strong></td>
</tr>
<tr>
<td></td>
<td>provider that hosts the control plane machines. This parameter value must</td>
<td></td>
</tr>
<tr>
<td></td>
<td>match the <strong>compute.platform</strong> parameter value.</td>
<td></td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is <strong>3</strong>, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td>dynamically tries to determine the capabilities of the provided credentials,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with a preference for mint mode on the platforms where multiple modes are</td>
<td></td>
</tr>
<tr>
<td></td>
<td>supported.</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use imageContentSources. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

Setting this field to Internal is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

** sshKey **

The SSH key or keys to authenticate access your cluster machines.

** NOTE **

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```
### Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.password</code></td>
<td>The password for the vCenter user name.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.datacenter</code></td>
<td>The name of the datacenter to use in the vCenter instance.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.defaultDatastore</code></td>
<td>The name of the default datastore to use for provisioning volumes.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.folder</code></td>
<td>Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the datacenter virtual machine folder.</td>
<td>String, for example, <code>/&lt;datacenter_name&gt;/vm/&lt;folder_name&gt;/&lt;subfolder_name&gt;</code>.</td>
</tr>
<tr>
<td><code>platform.vsphere.network</code></td>
<td>The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.cluster</code></td>
<td>The vCenter cluster to install the OpenShift Container Platform cluster in.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.apiVIP</code></td>
<td>The virtual IP (VIP) address that you configured for control plane API access.</td>
<td>An IP address, for example <code>128.0.0.1</code>.</td>
</tr>
<tr>
<td><code>platform.vsphere.ingressVIP</code></td>
<td>The virtual IP (VIP) address that you configured for cluster ingress.</td>
<td>An IP address, for example <code>128.0.0.1</code>.</td>
</tr>
</tbody>
</table>

11.3.8.1.5. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.clusterOSImage</code></td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <code>https://mirror.openshift.com/images/rhelcos&lt;version&gt;-vmware.&lt;architecture&gt;.ova</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>platform.vsphere.osDisk.diskSizeGB</code></td>
<td>The size of the disk in gigabytes.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.cpus</code></td>
<td>The total number of virtual processor cores to assign a virtual machine.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.coresPerSocket</code></td>
<td>The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is <code>platform.vsphere.cpus/platform.vsphere.coresPerSocket</code>. The default value is 1.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.memoryMB</code></td>
<td>The size of a virtual machine’s memory in megabytes.</td>
<td>Integer</td>
</tr>
</tbody>
</table>

### 11.3.8.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 3
    platform:
      vsphere:
        cpus: 2
        coresPerSocket: 2
        memoryMB: 8196
        osDisk:
          diskSizeGB: 120
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
  platform:
    vsphere:
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster
networking:
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The `controlPlane` section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to `Disabled`. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.

The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

11.3.8.3. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
     additionalTrustBundle: |
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
   ...
   ``

   - A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   - A proxy URL to use for creating HTTPS connections outside the cluster.
   - A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.
   - If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network
Operator then creates a **trusted-ca-bundle** config map that merges the contents specified for the **trustedCA** parameter with the RHCOS trust bundle. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil `spec`.

**NOTE**

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

### 11.3.9. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

**Phase 2**

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

### 11.3.10. Specifying advanced network configuration
You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**

- Create the `install-config.yaml` file and complete any modifications to it.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`
   
   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```bash
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```

   where:

   `<installation_directory>`
   
   Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

   **Specify a different VXLAN port for the OpenShift SDN network provider**

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   ```
4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

11.3.11. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the `Network` API in the `operator.openshift.io` API group.

The CNO configuration inherits the following fields during cluster installation from the `Network` API in the `Network.config.openshift.io` API group and these fields cannot be changed:

- **clusterNetwork**
  - IP address pools from which pod IP addresses are allocated.

- **serviceNetwork**
  - IP address pool for services.

- **defaultNetwork.type**
  - Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the `defaultNetwork` object in the CNO object named `cluster`.

11.3.11.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>metadata.name</code></td>
<td>string</td>
<td>The name of the CNO object. This name is always <code>cluster</code>.</td>
</tr>
<tr>
<td><code>spec.clusterNetwork</code></td>
<td>array</td>
<td>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.32.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the <code>install-config.yaml</code> file.</td>
</tr>
</tbody>
</table>
### Field | Type | Description
---|---|---
**spec.serviceNetwork**  | array  | A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:

```
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.

**spec.defaultNetwork**  | object  | Configures the Container Network Interface (CNI) cluster network provider for the cluster network.

**spec.kubeProxyConfig**  | object  | The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.

### defaultNetwork object configuration
The values for the `defaultNetwork` object are defined in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>type</strong></td>
<td>string</td>
<td>Either <strong>OpenShiftSDN</strong> or <strong>OVNKubernetes</strong>. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

#### NOTE
OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>openshiftSDNConfig</strong></td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td><strong>ovnKubernetesConfig</strong></td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

### Configuration for the OpenShift SDN CNI cluster network provider
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.
Table 11.19. openshiftSDNConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is <strong>NetworkPolicy</strong>. The values <strong>Multitenant</strong> and <strong>Subnet</strong> are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If your cluster requires different MTU values for different nodes, you must set this value to <strong>50</strong> less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of <strong>9001</strong>, and some have an MTU of <strong>1500</strong>, you must set this value to <strong>1450</strong>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>vxlanPort</td>
<td>integer</td>
<td>The port to use for all VXLAN packets. The default value is <strong>4789</strong>. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port <strong>9000</strong> and port <strong>9999</strong>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

Example OpenShift SDN configuration

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes CNI cluster network provider

```yaml
To configure OVN-Kubernetes CNI on OpenShift Container Platform, set:
```
The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

### Table 11.20. ovnKubernetesConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>genevePort</td>
<td>integer</td>
<td>The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

**Example OVN-Kubernetes configuration**

```yaml
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

**kubeProxyConfig object configuration**
The values for the `kubeProxyConfig` object are defined in the following table:

### Table 11.21. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

---

OpenShift Container Platform 4.6 Installing
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for <code>iptables</code> rules. The default value is 30s. Valid suffixes include <code>s</code>, <code>m</code>, and <code>h</code> and are described in the <code>Go time</code> package documentation.</td>
</tr>
</tbody>
</table>

**NOTE**

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the `iptablesSyncPeriod` parameter is no longer necessary.

<table>
<thead>
<tr>
<th>proxyArguments.iptables-min-sync-period</th>
<th>array</th>
<th>The minimum duration before refreshing <code>iptables</code> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include <code>s</code>, <code>m</code>, and <code>h</code> and are described in the <code>Go time</code> package. The default value is:</th>
</tr>
</thead>
</table>
|                                        |        | `kubeProxyConfig:
|                                        |        |   proxyArguments:
|                                        |        |     iptables-min-sync-period:
|                                        |        |       - 0s`                                                                                                                                   |

### 11.3.12. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:
For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

...  
INFO Install complete!  
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'  
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com  
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"  
INFO Time elapsed: 36m22s  

NOTE  
The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

IMPORTANT  
- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT  
You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.
11.3.13. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

11.3.13.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   
   ```
   $ tar xvzf <file>
   ```
5. Place the oc binary in a directory that is on your PATH.
   
   To check your PATH, execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

11.3.13.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the oc binary to a directory that is on your PATH.
   
   To check your PATH, open the command prompt and execute the following command:
   
   ```
   $ echo $PATH
   ```
OpenShift Container Platform 4.6 Installing

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 11.3.13.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the **Version** drop-down menu.
3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   $ oc <command>
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 11.3.14. Logging into the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the **kubeadmin** credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami

Example output
system:admin
```

11.3.15. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

11.3.15.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**.

**NOTE**

The Prometheus console provides an **ImageRegistryRemoved** alert, for example:

"Image Registry has been removed. **ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected. Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

11.3.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

11.3.15.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.
OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**
   
   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

**Example output**

```
storage:
pvc:
  claim: 1
```

Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.
4. Check the `clusteroperator` status:

   ```bash
   $ oc get clusteroperator image-registry
   ```

### 11.3.15.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only 1 replica:

   ```bash
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{
   "spec": {
   "rolloutStrategy": "Recreate",
   "replicas": 1
   }}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere `PersistentVolumeClaim` object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   **1** A unique name that represents the `PersistentVolumeClaim` object.

   **2** The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.

   **3** The access mode of the persistent volume claim. With `ReadWriteOnce`, the volume can be mounted with read and write permissions by a single node.

   **4** The size of the persistent volume claim.

   b. Create the `PersistentVolumeClaim` object from the file:
$ oc create -f pvc.yaml -n openshift-image-registry

3. Edit the registry configuration so that it references the correct PVC:

$ oc edit config.imageregistry.operator.openshift.io -o yaml

**Example output**

```yaml
storage:
pvc:
  claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

### 11.3.16. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

**Procedure**

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

### 11.3.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service
11.3.18. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

11.4. INSTALLING A CLUSTER ON VSPHERE WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure that you provision.

**IMPORTANT**

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

11.4.1. Prerequisites

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide ReadWriteMany access modes.
- Review details about the OpenShift Container Platform installation and update processes.
- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.
- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

11.4.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
• Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 11.4.3. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

#### Table 11.22. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the <a href="https://access.redhat.com/documentation/en-US/Red-Hat-Enterprise-Linux/8/html/virtualization/Hypervisor-Glossary">Red Hat Enterprise Linux 8 supported hypervisors list</a></td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See [Edit Time Configuration for a Host](https://vmware.com) in the VMware documentation.

### 11.4.4. Machine requirements for a cluster with user-provisioned infrastructure

1738
For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

11.4.4.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

**IMPORTANT**

All virtual machines must reside in the same datastore and in the same folder as the installer.

11.4.4.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

11.4.4.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

|---------|------------------|---------|-------------|---------|----------|

---

[1] vCPU
[2] IOPS
1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: $(\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}$.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

### Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

### Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.

2. Provision the required load balancers.

3. Configure the ports for your machines.

4. Configure DNS.

5. Ensure network connectivity.

### Networking requirements for user-provisioned infrastructure

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>
All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 11.23. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 11.24. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 11.25. Control plane machines to control plane machines
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 11.26. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**Ethernet adaptor hardware address requirements**

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- 00:05:69:00:00:00 to 00:05:69:FF:FF
- 00:0c:29:00:00:00 to 00:0c:29:FF:FF
- 00:1c:14:00:00:00 to 00:1c:14:FF:FF
- 00:50:56:00:00:00 to 00:50:56:FF:FF
If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**Additional resources**

- Configuring chrony time service

### 11.4.5.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td><code>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
### Component | Record | Description
--- | --- | ---
Routes | *.apps.<cluster_name>.<base_domain>. | Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Bootstrap | bootstrap.<cluster_name>.<base_domain>. | Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

Master hosts | <master><n>.<cluster_name>.<base_domain>. | DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts | <worker><n>.<cluster_name>.<base_domain>. | Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**TIP**
You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

```bash
$TTL 1W
@ IN SOA ns1.example.com. root (2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;```
The following example BIND zone file shows sample PTR records for reverse name resolution.

**Example 11.8. Sample DNS zone database for reverse records**

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

11.4.6. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
   ``

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
   
   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.
3. Add your SSH private key to the **ssh-agent**:

```bash
$ ssh-add <path>/<file_name>
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

**11.4.7. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.4.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

11.4.8.1. Sample install-config.yaml file for VMware vSphere

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

   ```yaml
   apiVersion: v1
   ```
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.</td>
</tr>
<tr>
<td>2, 5</td>
<td>The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.</td>
</tr>
<tr>
<td>3, 6</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.</td>
</tr>
<tr>
<td>7</td>
<td>The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.</td>
</tr>
</tbody>
</table>
8. The cluster name that you specified in your DNS records.

9. The fully-qualified hostname or IP address of the vCenter server.

10. The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

11. The password associated with the vSphere user.

12. The vSphere datacenter.

13. The default vSphere datastore to use.

14. Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, 

   /<datacenter_name>/vm/<folder_name>/<subfolder_name>. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

15. Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

   **IMPORTANT**

   The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

16. The pull secret that you obtained from OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

17. The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

11.4.8.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the install-config.yaml file.

**Prerequisites**

- You have an existing install-config.yaml file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object's spec.noProxy field to bypass the proxy if necessary.
NOTE

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...
   ```

   - **1** A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   - **2** A proxy URL to use for creating HTTPS connections outside the cluster.
   - **3** A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.
   - **4** If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RH COS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RH COS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

11.4.9. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program.
- You created the `install-config.yaml` installation configuration file.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   `$/openshift-install create manifests --dir <installation_directory>`

   **Example output**

   ```
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.
2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```sh
   $ rm -f openshift/99_openshift-cluster-api_master-machines-* .yaml
   openshift/99_openshift-cluster-api_worker-machineset-* .yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.

   - You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the `mastersSchedulable` parameter in the
   `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

   c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```sh
   $ ./openshift-install create ignition-configs --dir <installation_directory>
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   - auth
     - kubeadm-password
     - kubeconfig
   - bootstrap.ign
   - master.ign
   - metadata.json
   - worker.ign

11.4.10. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.
Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  ```
  $ jq -r .infraID <installation_directory>/metadata.json
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6
```

The output of this command is your cluster name and a random string.

11.4.11. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a vSphere cluster.

Procedure

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

```json
{
  "ignition": {
    "config": {
      "merge": [
      {
        "source": "<bootstrap_ignition_config_url>",
        "verification": {}
      }
      ]
    },
    "timeouts": {},
    "version": "3.1.0"
  },
```

CHAPTER 11. INSTALLING ON VSPHERE
Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:
   - `<installation_directory>/master.ign`
   - `<installation_directory>/worker.ign`
   - `<installation_directory>/merge-bootstrap.ign`

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

   ```bash
   $ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
   $ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
   $ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
   ```

   **IMPORTANT**
   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

   **IMPORTANT**
   The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

   The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   a. Click the **VMs and Templates** view.
b. Right-click the name of your datacenter.

c. Click **New Folder → New VM and Template Folder**.

d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the **install-config.yaml** file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.

7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

   **NOTE**

   In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

   a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.

   b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.

   c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.

   d. On the **Select a compute resource** tab, click the name of your vSphere cluster.

   e. On the **Select storage** tab, configure the storage options for your VM.

      - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.

      - Select the datastore that you specified in your **install-config.yaml** file.

   f. On the **Select network** tab, specify the network that you configured for the cluster, if available.

   g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

   **IMPORTANT**

   Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.

   a. Right-click the template name and click **Clone → Clone to Virtual Machine**.

   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.
c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.

d. On the **Select a compute resource** tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

e. Optional: On the **Select storage** tab, customize the storage options.

f. On the **Select clone options**, select **Customize this virtual machine’s hardware**

g. On the **Customize hardware** tab, click **VM Options → Advanced**.
   - Optional: Override default DHCP networking in vSphere. To enable static IP networking:
     i. Set your static IP configuration:

```
$ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>::none
     nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

**Example command**

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
     nameserver=8.8.8.8"
```

   ii. Set the **guestinfo.afterburn.initrd.network-kargs** property before booting a VM from an OVA in vSphere:

```
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

   - Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:
     - Memory reservation value must be equal to its configured memory size.
     - CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.

   - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
     - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
     - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
     - **disk.EnableUUID**: Specify **TRUE**.

h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
i. Complete the configuration and power on the VM.

9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

11.4.12. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

**Prerequisites**

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

**Procedure**

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click **Clone → Clone to Virtual Machine**
   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.
   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
   d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
   e. Optional: On the **Select storage** tab, customize the storage options.
   f. On the **Select clone options**, select **Customize this virtual machine’s hardware**
   g. On the **Customize hardware** tab, click **VM Options → Advanced**.
      - From the **Latency Sensitivity** list, select **High**.
      - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
        - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
        - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
        - **disk.EnableUUID**: Specify **TRUE**.
   h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values
as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under **Add network adapter** if there are multiple networks available.

i. Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

### 11.4.13. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- **Create separate partitions**: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making `/var` or a subdirectory of `/var`, such as `/var/lib/etcd`, a separate partition, but not both.

  **IMPORTANT**
  
  Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- **Retain existing partitions**: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

**Creating a separate `/var` partition**

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- **`/var/lib/containers`**: Holds container-related content that can grow as more images and containers are added to a system.

- **`/var/lib/etcd`**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.

- **`/var`**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.
Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadmin-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...
   ```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
       machineconfiguration.openshift.io/role: worker
     name: 98-var-partition
   spec:
     config:
       ignition:
         version: 3.1.0
       storage:
         disks:
           - device: /dev/<device_name> ¹
             partitions:
               - label: var
                 startMiB: <partition_start_offset> ²
                 sizeMiB: <partition_size> ³
             filesystems:
               - device: /dev/disk/by-partlabel/var
                 path: /var
                 format: xfs
         systemd:
           units:
             - name: var.mount ⁴
               enabled: true
               contents: |
               [Unit]
                 Before=local-fs.target
               [Mount]
                 What=/dev/disk/by-partlabel/var
                 Where=/var
   ```
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.

The prjquota mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 11.4.14. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

### 11.4.14.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:
   
   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:
   
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 11.4.14.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
   
   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 11.4.14.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.

4. Unpack and unzip the archive.

5. Move the oc binary to a directory on your PATH. To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

### 11.4.15. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**

1. Monitor the bootstrap process:

   ```
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete --log-level=info
   ```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**Example output**

   ```
   INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
   INFO API v1.19.0 up
   INFO Waiting up to 30m0s for bootstrapping to complete...
   INFO It is now safe to remove the bootstrap resources
   ```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.
2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

**IMPORTANT**

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

### 11.4.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ``

   ![Image 1](image.png)
   
   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ``

   **Example output**

   `system:admin`

### 11.4.17. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   ```
Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the Pending or Approved status for each machine that you added to the cluster:

   $ oc get csr

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vnps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
</tbody>
</table>

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in Pending status, approve the CSRs for your cluster machines:

   NOTE

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the machine-approver if the Kubelet requests a new certificate with identical parameters.
CHAPTER 11. INSTALLING ON VSPHERE

NOTE
For clusters running on platforms that are not machine API enabled, such as bare
metal and other user-provisioned infrastructure, you must implement a method
of automatically approving the kubelet serving certificate requests (CSRs). If a
request is not approved, then the oc exec, oc rsh, and oc logs commands
cannot succeed, because a serving certificate is required when the API server
connects to the kubelet. Any operation that contacts the Kubelet endpoint
requires this certificate approval to be in place. The method must watch for new
CSRs, confirm that the CSR was submitted by the node-bootstrapper service
account in the system:node or system:admin groups, and confirm the identity
of the node.
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

To approve all pending CSRs, run the following command:
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve

NOTE
Some Operators might not become available until some CSRs are approved.
4. Now that your client requests are approved, you must review the server requests for each
machine that you added to the cluster:
$ oc get csr

Example output
NAME
AGE REQUESTOR
CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for
your cluster machines:
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

1767


To approve all pending CSRs, run the following command:

```shell
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```shell
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).

### 11.4.18. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

```shell
$ watch -n5 oc get clusteroperators
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
</tbody>
</table>
2. Configure the Operators that are not available.

11.4.18.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

11.4.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.
Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

### 11.4.18.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended. Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is **emptyDir**, the replica number cannot be greater than 1.
3. Check the registry configuration:

   $ oc edit configs.imageregistry.operator.openshift.io

**Example output**

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage PVC`.

4. Check the `clusteroperator` status:

   $ oc get clusteroperator image-registry

**11.4.18.2.2. Configuring storage for the image registry in non-production clusters**

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

  ```
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage": {"emptyDir": {}}}}'
  ```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

| Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found |

Wait a few minutes and run the command again.

**11.4.18.2.3. Configuring block registry storage for VMware vSphere**

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.
IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only 1 replica:

   ```bash
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy":"Recreate","replicas":1}}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
   
   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   1. A unique name that represents the **PersistentVolumeClaim** object.
   2. The namespace for the **PersistentVolumeClaim** object, which is `openshift-image-registry`.
   3. The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
   4. The size of the persistent volume claim.

   b. Create the **PersistentVolumeClaim** object from the file:

   ```bash
   $ oc create -f pvc.yaml -n openshift-image-registry
   ```

3. Edit the registry configuration so that it references the correct PVC:

   ```bash
   $ oc edit config.imageregistry.operator.openshift.io -o yaml
   ```

Example output
Creating a custom PVC allows you to leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see Configuring the registry for vSphere.

**11.4.19. Completing installation on user-provisioned infrastructure**

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   ```shell
   $ watch -n5 oc get clusteroperators
   ```

**Example output**

```text
<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>
```
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.
   
   a. To view a list of all pods, use the following command:

   ```
   $ oc get pods --all-namespaces
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTARTS AGE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td></td>
</tr>
<tr>
<td>Running</td>
<td>1</td>
<td>9m</td>
<td></td>
</tr>
</tbody>
</table>
b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```bash
$ oc logs <pod_name> -n <namespace>  
```

1 Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.

### 11.4.20. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

**Procedure**

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

### 11.4.21. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.
11.5. INSTALLING A CLUSTER ON VSPHERE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure that you provision with customized network configuration options. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only kubeProxy configuration parameters in a running cluster.

**IMPORTANT**

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

11.5.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.

- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. Verify that port 443 is accessible.

- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.

- If you use a firewall, you must configure it to access Red Hat Insights.

11.5.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 11.5.3. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 11.29. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See *Edit Time Configuration for a Host* in the VMware documentation.
11.5.4. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

11.5.4.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

**IMPORTANT**

All virtual machines must reside in the same datastore and in the same folder as the installer.

11.5.4.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

11.5.4.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

### 11.5.4.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

### 11.5.5. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

### 11.5.5.1. Networking requirements for user-provisioned infrastructure
All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

### Table 11.30. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

### Table 11.31. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

### Table 11.32. Control plane machines to control plane machines
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

Important
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. API load balancer. Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   Important
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

Table 11.33. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>HTTPS traffic</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>80</td>
<td>HTTP traffic</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**Ethernet adaptor hardware address requirements**

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- 00:05:69:00:00:00 to 00:05:69:FF:FF:FF
- 00:0C:29:00:00:00 to 00:0C:29:FF:FF:FF
- 00:1C:14:00:00:00 to 00:1C:14:FF:FF:FF
- 00:50:56:00:00:00 to 00:50:56:FF:FF:FF
If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**Additional resources**

- Configuring chrony time service

### 11.5.5.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

**Table 11.35. Required DNS records**

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
<td></td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
Routes

*.apps.<cluster_name>.<base_domain>.

Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Bootstrap

bootstrap.<cluster_name>.<base_domain>.

Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.

Master hosts

<master><n>.
<cluster_name>.
<base_domain>.

DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts

<worker><n>.
<cluster_name>.
<base_domain>.

Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

```
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 11.10. Sample DNS zone database for reverse records

$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial
3H ; refresh (3 hours)
30M ; retry (30 minutes)
2W ; expiry (2 weeks)
1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF

11.5.6. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name> 1
   ```

   ![Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.](image)

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.
1. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> ①
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

① Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

### 11.5.7. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```
5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

11.5.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

11.5.8.1. Sample `install-config.yaml` file for VMware vSphere

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.
9 The fully-qualified hostname or IP address of the vCenter server.

10 The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

11 The password associated with the vSphere user.

12 The vSphere datacenter.

13 The default vSphere datastore to use.

14 Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

16 The pull secret that you obtained from OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

17 The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

11.5.8.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.
NOTE

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
     noProxy: example.com  # 3
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
   ...
   ``

   1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

   2. A proxy URL to use for creating HTTPS connections outside the cluster.

   3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter's IP address and the IP range that you use for its machines.

   4. If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RH COS trust bundle. The `additionalTrustBundle` field is required unless the proxy's identity certificate is signed by an authority from the RH COS trust bundle.

NOTE

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.
The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 11.5.9. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

**Phase 2**

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.

### 11.5.10. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**
• Create the `install-config.yaml` file and complete any modifications to it.
• Create the Ignition config files for your cluster.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```sh
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`
   
   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```sh
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```

   where:

   `<installation_directory>`
   
   Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

   **Specify a different VXLAN port for the OpenShift SDN network provider**

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
     defaultNetwork:
       openshiftSDNConfig:
         vxlanPort: 4800
   ```

4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

6. Remove the Kubernetes manifest files that define the control plane machines and compute machineSets:
Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the MachineSet files to create compute machines by using the machine API, but you must update references to them to match your environment.

### 11.5.11. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the Network API in the `operator.openshift.io` API group.

The CNO configuration inherits the following fields during cluster installation from the Network API in the `Network.config.openshift.io` API group and these fields cannot be changed:

- **clusterNetwork**
  - IP address pools from which pod IP addresses are allocated.

- **serviceNetwork**
  - IP address pool for services.

- **defaultNetwork.type**
  - Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the `defaultNetwork` object in the CNO object named `cluster`.

#### 11.5.11.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>metadata.name</code></td>
<td>string</td>
<td>The name of the CNO object. This name is always <code>cluster</code>.</td>
</tr>
</tbody>
</table>
| `spec.clusterNetwork`| array  | A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:

```yaml
spec:
  clusterNetwork:
    - cidr: 10.128.0.0/19
      hostPrefix: 23
    - cidr: 10.128.32.0/19
      hostPrefix: 23
```

- This value is ready-only and specified in the `install-config.yaml` file.
A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:

```
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.

**defaultNetwork object configuration**

The values for the `defaultNetwork` object are defined in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>type</code></td>
<td>string</td>
<td>Either <code>OpenShiftSDN</code> or <code>OVNKubernetes</code>. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td><code>openshiftSDNConfig</code></td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider by default.</td>
</tr>
<tr>
<td><code>ovnKubernetesConfig</code></td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

**Configuration for the OpenShift SDN CNI cluster network provider**

The following table describes the configuration fields for the OpenShift SDN CNI cluster network provider:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>serviceNetwork</code></td>
<td>array</td>
<td>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:</td>
</tr>
</tbody>
</table>

```
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

Table 11.38. openshiftSDNConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy. The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>vxlanPort</td>
<td>integer</td>
<td>The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation. If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number. On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999.</td>
</tr>
</tbody>
</table>

Example OpenShift SDN configuration

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```
Configuration for the OVN-Kubernetes CNI cluster network provider
The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

Table 11.39. ovnKubernetesConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| mtu         | integer   | The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.  
If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.  
If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.  
This value cannot be changed after cluster installation. |
| genevePort  | integer   | The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.                                                                                     |

Example OVN-Kubernetes configuration

```yaml
defaultNetwork:  
type: OVNKubernetes  

ovnKubernetesConfig:  
mtu: 1400  
genevePort: 6081
```

kubeProxyConfig object configuration
The values for the kubeProxyConfig object are defined in the following table:

Table 11.40. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
### iptablesSyncPeriod

String

The refresh period for iptables rules. The default value is 30s. Valid suffixes include s, m, and h and are described in the Go time package documentation.

**NOTE**

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the iptablesSyncPeriod parameter is no longer necessary.

---

### proxyArguments.iptables-min-sync-period

Array

The minimum duration before refreshing iptables rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include s, m, and h and are described in the Go time package. The default value is:

    kubeProxyConfig:
    proxyArguments:
      iptables-min-sync-period:
        - 0s

#### 11.5.12. Creating the Ignition config files

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

- Obtain the Ignition config files:

```bash
$ ./openshift-install create ignition-configs --dir <installation_directory>
```

1. For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

If you created an `install-config.yaml` file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

The following files are generated in the directory:

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

**11.5.13. Extracting the infrastructure name**

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

- You generated the Ignition config files for your cluster.

- You installed the `jq` package.

Procedure
To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

```
$ jq -r .infraID <installation_directory>/metadata.json
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
openshift-vw9j6
```

The output of this command is your cluster name and a random string.

### 11.5.14. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

**Prerequisites**

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a [vSphere cluster](#).

**Procedure**

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

```json
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>",
          "verification": {}
        }
      ],
      "timeouts": {},
      "version": "3.1.0"
    },
    "networkd": {},
    "passwd": {}
  }
}
```
1. Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:
   - `<installation_directory>/master.ign`
   - `<installation_directory>/worker.ign`
   - `<installation_directory>/merge-bootstrap.ign`

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

   ```bash
   $ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
   $ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
   $ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
   ```

   **IMPORTANT**
   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

   **IMPORTANT**
   The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

   The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   a. Click the **VMs and Templates** view.
   b. Right-click the name of your datacenter.
c. Click **New Folder → New VM and Template Folder**.

d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the `install-config.yaml` file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.

7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

   **NOTE**

   In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

   a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.

   b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.

   c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as `Template-RHCOS`. Click the name of your vSphere cluster and select the folder you created in the previous step.

   d. On the **Select a compute resource** tab, click the name of your vSphere cluster.

   e. On the **Select storage** tab, configure the storage options for your VM.

      - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.

      - Select the datastore that you specified in your `install-config.yaml` file.

   f. On the **Select network** tab, specify the network that you configured for the cluster, if available.

   g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

   **IMPORTANT**

   Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.

   a. Right-click the template name and click **Clone → Clone to Virtual Machine**

   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as `control-plane-0` or `compute-1`.

   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
d. On the **Select a compute resource** tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

e. Optional: On the **Select storage** tab, customize the storage options.

f. On the **Select clone options**, select **Customize this virtual machine’s hardware**

h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.

i. Complete the configuration and power on the VM.

---

**Example command**

```bash
$ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

```bash
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

```bash
$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

---

Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

- Memory reservation value must be equal to its configured memory size.
- CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:

  i. Set your static IP configuration:

  ```bash
  $ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>:none
  nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
  ```

  ii. Set the **guestinfo.afterburn.initrd.network-kargs** property before booting a VM from an OVA in vSphere:

  ```bash
  $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
  ```

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

  ```bash
  $ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
  nameserver=8.8.8.8"
  ```

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

  ```bash
  $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
  ```

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

  ```bash
  $ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
  nameserver=8.8.8.8"
  ```

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

  ```bash
  $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
  ```

- Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:

  - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.

  - **guestinfo.ignition.config.data.encoding**: Specify **base64**.

  - **disk.EnableUUID**: Specify **TRUE**.
9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

11.5.15. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

**Prerequisites**

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

**Procedure**

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click **Clone → Clone to Virtual Machine**
   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.
   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
   d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
   e. Optional: On the **Select storage** tab, customize the storage options.
   f. On the **Select clone options**, select **Customize this virtual machine’s hardware**
   g. On the **Customize hardware** tab, click **VM Options → Advanced**.
      - From the **Latency Sensitivity** list, select **High**.
      - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
         - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
         - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
         - **disk.EnableUUID**: Specify **TRUE**.
   h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under **Add network adapter** if there are multiple networks available.
i. Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

11.5.16. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making /var or a subdirectory of /var, such as /var/lib/etcd, a separate partition, but not both.

  **IMPORTANT**

  Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to coreos-installer that allow you to retain existing data partitions.

Creating a separate /var partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the /var partition or a subdirectory of /var. For example:

- /var/lib/containers: Holds container-related content that can grow as more images and containers are added to a system.
- /var/lib/etcd: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- /var: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a /var directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because /var must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate /var partition by creating a machine config that is inserted during the openshift-install preparation phases of an OpenShift Container Platform installation.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:
2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

```
$ mkdir $HOME/clusterconfig
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
  99_kubeadmin-password-secret.yaml
  99_openshift-cluster-api_master-machines-0.yaml
  99_openshift-cluster-api_master-machines-1.yaml
  99_openshift-cluster-api_master-machines-2.yaml

apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
  storage:
    disks:
      - device: /dev/<device_name>
        partitions:
          - label: var
            startMiB: <partition_start_offset>
            sizeMiB: <partition_size>
    filesystems:
      - device: /dev/disk/by-partlabel/var
        path: /var
        format: xfs
    systemd:
      units:
        - name: var.mount
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota
            [Install]
            WantedBy=local-fs.target
```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future re-installs of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.

The priquota mount option must be enabled for filesystems used for container storage.

NOTE
When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

11.5.17. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

**IMPORTANT**

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

11.5.18. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   1

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```
11.5.19. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   ```
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  63m  v1.19.0
   master-1  Ready     master  63m  v1.19.0
   master-2  Ready     master  64m  v1.19.0
   ```

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```
   $ oc get csr
   ```

   **Example output**

   ```
   NAME        AGE     REQUESTOR                                                                   CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   ... 
   ```

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.
3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the machine-approver if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name> 1
  
  1 <csr_name> is the name of a CSR from the list of current CSRs.
  ```

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```bash
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
</tbody>
</table>
5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

   - To approve them individually, run the following command for each valid CSR:
     
     ```bash
     $ oc adm certificate approve <csr_name>  
     
     <csr_name> is the name of a CSR from the list of current CSRs.
     
     - To approve all pending CSRs, run the following command:
     
     ```bash
     $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
     ```

6. After all client and server CSRs have been approved, the machines have the Ready status. Verify this by running the following command:

   ```bash
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  73m  v1.20.0
   master-1  Ready     master  73m  v1.20.0
   master-2  Ready     master  74m  v1.20.0
   worker-0  Ready     worker  11m  v1.20.0
   worker-1  Ready     worker  11m  v1.20.0
   ```

   **NOTE**

   It can take a few minutes after approval of the server CSRs for the machines to transition to the Ready status.

**Additional information**

- For more information on CSRs, see Certificate Signing Requests.

### 11.5.19.1. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**
1. Watch the cluster components come online:

```bash
$ watch -n5 oc get clusteroperators
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

### 11.5.19.2. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**.
NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

11.5.19.3. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

11.5.19.3.1. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the Recreate rollout strategy.

IMPORTANT

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

Procedure

1. To set the image registry storage as a block storage type, patch the registry so that it uses the Recreate rollout strategy and runs with only 1 replica:

   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere PersistentVolumeClaim object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
   ```
resources:
  requests:
  
storage: 100Gi

1. A unique name that represents the `PersistentVolumeClaim` object.

2. The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.

3. The access mode of the persistent volume claim. With `ReadWriteOnce`, the volume can be mounted with read and write permissions by a single node.

4. The size of the persistent volume claim.

b. Create the `PersistentVolumeClaim` object from the file:

   $ oc create -f pvc.yaml -n openshift-image-registry

3. Edit the registry configuration so that it references the correct PVC:

   $ oc edit config.imageregistry.operator.openshift.io -o yaml

   **Example output**

   ```yaml
   storage:
   pvc:
     claim: 1
   ```

   1. Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

### 11.5.20. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   $ watch -n5 oc get clusteroperators

   **Example output**
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.
   a. To view a list of all pods, use the following command:

   $ oc get pods --all-namespaces

   **Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2m</td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5m</td>
</tr>
</tbody>
</table>

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   $ oc logs <pod_name> -n <namespace>

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.
11.5.21. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

11.5.22. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

11.5.23. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

11.6. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK

In OpenShift Container Platform 4.6, you can install a cluster on VMware vSphere infrastructure in a restricted network by creating an internal mirror of the installation release content.

11.6.1. Prerequisites

- Create a registry on your mirror host and obtain the imageContentSources data for your version of OpenShift Container Platform.
IMPORTANT

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide the ReadWriteMany access mode.

- Review details about the OpenShift Container Platform installation and update processes.

- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.

- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.

- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

NOTE

If you are configuring a proxy, be sure to also review this site list.

11.6.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

11.6.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The ClusterVersion status includes an Unable to retrieve available updates error.

- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

11.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.
You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access **Quay.io** to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

11.6.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 11.41. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.
IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

11.6.5. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 11.11. Roles and privileges required for installation

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
</table>
| vSphere vCenter               | Always        | Cns.Searchable
InventoryService.Tagging.AttachTag
InventoryService.Tagging.CreateCategory
InventoryService.Tagging.CreateTag
InventoryService.Tagging.DeleteCategory
InventoryService.Tagging.DeleteTag
InventoryService.Tagging>EditCategory
InventoryService.Tagging>EditTag
Sessions.ValidateSession
StorageProfile.View |
| vSphere vCenter Cluster       | Always        | Host.Config.Storage
Resource.AssignVMToPool
VApp.AssignResourcePool
VApp.Import
VirtualMachine.Config.AddNewDisk |
<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 11.12. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td>Installation program creates the folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.

Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.
Cluster resources
When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits
Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements
You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. The VM in your restricted network must have access to vCenter so that it can provision and manage nodes, persistent volume claims (PVCs), and other resources. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

NOTE
It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses
An installer-provisioned vSphere installation requires two static IP addresses:

- The API address is used to access the cluster API.
- The Ingress address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.
DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 11.42. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

11.6.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE
In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure
1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:
Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

```
$ ssh-keygen -t ed25519 -N "\n-f <path>/<file_name>
```

1. Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

2. Start the ssh-agent process as a background task:

```
$ eval "$(ssh-agent -s)"
```

**Example output**

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name>
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1. Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**11.6.7. Adding vCenter root CA certificates to your system trust**

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.
Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.

2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    └── 7e757f6a.0.crt
```

3 directories, 15 files

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

11.6.8. Creating the RHCOS image for restricted network installations

Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network VMware vSphere environment.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

**Procedure**

2. Under **Version**, select the most recent release of OpenShift Container Platform 4.6 for RHEL 8.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - vSphere** image.

4. Upload the image you downloaded to a location that is accessible from the bastion server.

The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

### 11.6.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

- Have the **imageContentSources** values that were generated during mirror registry creation.

- Obtain the contents of the certificate for your mirror registry.

- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.

**Procedure**

1. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

      ```bash
      $ ./openshift-install create install-config --dir <installation_directory> ①
      ```

      ① For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select vsphere as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.

The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. In the `install-config.yaml` file, set the value of `platform.vsphere.clusterOSImage` to the image location or name. For example:

   ```yaml
   platform:
     vsphere:
   ```
clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-vmware.x86_64.ova?
sha256=f8eb6d68e8a1f2a454c19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d

3. Edit the **install-config.yaml** file to provide the additional information that is required for an installation in a restricted network.
   a. Update the pullSecret value to contain the authentication information for your registry:

   ```yaml
   pullSecret: {"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>", "email": "you@example.com"}}}
   ``

   For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

   b. Add the **additionalTrustBundle** parameter and value.

   ```yaml
   additionalTrustBundle: |
   -----BEGIN CERTIFICATE-----
   ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
   -----END CERTIFICATE-----
   ``

   The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

   c. Add the image content resources, which look like this excerpt:

   ```yaml
   imageContentSources:
   - mirrors:
     - `<mirror_host_name>:5000/<repo_name>/release`
       source: quay.example.com/openshift-release-dev/ocp-release
     - mirrors:
       - `<mirror_host_name>:5000/<repo_name>/release`
       source: registry.example.com/ocp/release
   ``

   To complete these values, use the **imageContentSources** that you recorded during mirror registry creation.

4. Make any other modifications to the **install-config.yaml** file that you require. You can find more information about the available parameters in the **Installation configuration parameters** section.

5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 11.6.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe
your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 11.6.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;.&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}</code>. <code>{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere.** For additional information about `platform.<platform>` parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: <strong>aws, baremetal, azure, openstack, ovirt, vsphere.</strong> For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td><code>{  &quot;auths&quot;:{    &quot;cloud.openshift.com&quot;:{      &quot;auth&quot;:&quot;b3Blb=&quot;,      &quot;email&quot;:&quot;you@example.com&quot;    },    &quot;quay.io&quot;:{      &quot;auth&quot;:&quot;b3Blb=&quot;,      &quot;email&quot;:&quot;you@example.com&quot;    }  }}</code></td>
</tr>
</tbody>
</table>

### 11.6.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 11.44. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.network Type</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods. The default value is 10.128.0.0/14 with a host prefix of /23. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 ((2^{32} - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>An array with an IP address block in CIDR format. For example:</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>
**networking.machineNetwork.cidr**

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

### 11.6.9.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

#### Table 11.45. Optional parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or *hyperthreading*, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hyperthreading</em>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architect</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>controlPlane.hypertreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td><code>controlPlane.name</code></td>
<td>Required if you use <code>controlPlane</code>. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td><code>controlPlane.platform</code></td>
<td>Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td><code>controlPlane.replicas</code></td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td><strong>IMPORTANT</strong></td>
<td>The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the <strong>x86_64</strong> architecture.</td>
<td></td>
</tr>
<tr>
<td><strong>NOTE</strong></td>
<td>If you are using Azure File storage, you cannot enable FIPS mode.</td>
<td></td>
</tr>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
### Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table:

**Table 11.46. Additional VMware vSphere cluster parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.vCenter</td>
<td>The fully-qualified hostname or IP address of the vCenter server.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.username</td>
<td>The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>platform.vsphere.password</td>
<td>The password for the vCenter user name.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.datacenter</td>
<td>The name of the datacenter to use in the vCenter instance.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.defaultDatastore</td>
<td>The name of the default datastore to use for provisioning volumes.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.folder</td>
<td>Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the datacenter virtual machine folder.</td>
<td>String, for example, /&lt;datacenter_name&gt;/vm/&lt;folder_name&gt;/&lt;subfolder_name&gt;.</td>
</tr>
<tr>
<td>platform.vsphere.network</td>
<td>The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.cluster</td>
<td>The vCenter cluster to install the OpenShift Container Platform cluster in.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.apiVIP</td>
<td>The virtual IP (VIP) address that you configured for control plane API access.</td>
<td>An IP address, for example 128.0.0.1</td>
</tr>
<tr>
<td>platform.vsphere.ingressVIP</td>
<td>The virtual IP (VIP) address that you configured for cluster ingress.</td>
<td>An IP address, for example 128.0.0.1</td>
</tr>
</tbody>
</table>

11.6.9.15. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

Table 11.47. Optional VMware vSphere machine pool parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.clusterOSImage</td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <a href="https://mirror.openshift.com/images/rhcos-">https://mirror.openshift.com/images/rhcos-</a>&lt;version&gt;-vmware.&lt;architecture&gt;.ova.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>platform.vsphere.osDisk.diskSizeGB</td>
<td>The size of the disk in gigabytes.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.cpus</td>
<td>The total number of virtual processor cores to assign a virtual machine.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.coresPerSocket</td>
<td>The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket. The default value is 1</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.memoryMB</td>
<td>The size of a virtual machine’s memory in megabytes.</td>
<td>Integer</td>
</tr>
</tbody>
</table>

11.6.9.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com  # 1
compute:  # 2
- hyperthreading: Enabled  # 3
  name: worker
  replicas: 3
  platform:
    vsphere:  # 4
      cpus: 2
      coresPerSocket: 2
      memoryMB: 8196
      osDisk:
        diskSizeGB: 120
  controlPlane:  # 5
    hyperthreading: Enabled  # 6
    name: master
    replicas: 3
    platform:
      vsphere:  # 7
        cpus: 4
        coresPerSocket: 2
        memoryMB: 16384
        osDisk:
          diskSizeGB: 120
  metadata:
    name: cluster  # 8
    platform:
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.

The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program
The location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that is accessible from the bastion server.

For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

Provide the contents of the certificate file that you used for your mirror registry.

Provide the `imageContentSources` section from the output of the command to mirror the repository.

### 11.6.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

#### Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

#### Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE
The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE
Only the Proxy object named cluster is supported, and no additional proxies can be created.

11.6.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure
1. Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
```

1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

...  INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export 
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s

**NOTE**

The cluster access and credential information also outputs to
`<installation_directory>/.openshift_install.log` when an installation succeeds.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.
**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

### 11.6.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

#### 11.6.11.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the OpenShift v4.6 Linux Client entry and save the file.
4. Unpack the archive:
   
   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```  

#### 11.6.11.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click **Download Now** next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   C:\> oc <command>
   ```

11.6.11.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

11.6.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

**Procedure**

1. Export the kubeadmin credentials:
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```bash
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

Example output

```
system:admin
```

### 11.6.13. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

```bash
$ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the `Administration → Cluster Settings → Global Configuration → OperatorHub` page, click the `Sources` tab, where you can create, delete, disable, and enable individual sources.

### 11.6.14. Creating registry storage

After you install the cluster, you must create storage for the Registry Operator.

#### 11.6.14.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as `Removed`. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from `Removed` to `Managed`. 
NOTE

The Prometheus console provides an `ImageRegistryRemoved` alert, for example:

"Image Registry has been removed. `ImageStreamTags`, `BuildConfigs` and `DeploymentConfigs` which reference `ImageStreamTags` may not work as expected. Please configure storage and update the config to `Managed` state by editing `configs.imageregistry.operator.openshift.io`.

11.6.14.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

11.6.14.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

IMPORTANT

OpenShift Container Platform supports `ReadWriteOnce` access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, `ReadWriteMany` access is required.

- Must have "100Gi" capacity.

IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.
Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```
   $ oc get clusteroperator image-registry
   ```

11.6.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See About remote health monitoring for more information about the Telemetry service

11.6.16. Next steps
11.7. INSTALLING A CLUSTER ON VSPHERE IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure that you provision in a restricted network.

IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the vSphere platform and the installation process of OpenShift Container Platform. Use the user-provisioned infrastructure installation instructions as a guide; you are free to create the required resources through other methods.

11.7.1. Prerequisites

- Create a registry on your mirror host and obtain the `imageContentSources` data for your version of OpenShift Container Platform.

  IMPORTANT

  Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- Provision persistent storage for your cluster. To deploy a private image registry, your storage must provide `ReadWriteMany` access modes.

- Review details about the OpenShift Container Platform installation and update processes.

- Completing the installation requires that you upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA on vSphere hosts. The machine from which you complete this process requires access to port 443 on the vCenter and ESXi hosts. You verified that port 443 is accessible.

- If you use a firewall, you confirmed with the administrator that port 443 is accessible. Control plane nodes must be able to reach vCenter and ESXi hosts on port 443 for the installation to succeed.

- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

  NOTE

  Be sure to also review this site list if you are configuring a proxy.

11.7.2. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active
connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

**IMPORTANT**

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

### 11.7.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The `ClusterVersion` status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

### 11.7.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access [OpenShift Cluster Manager](https://github.com/openshift) to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access [Quay.io](https://quay.io) to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.
11.7.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

Table 11.48. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
<tr>
<td>Optional: Networking (NSX-T)</td>
<td>vSphere 6.5U3 or vSphere 6.7U2 and later</td>
<td>vSphere 6.5U3 or vSphere 6.7U2+ are required for OpenShift Container Platform. VMware’s NSX Container Plug-in (NCP) 3.0.2 is certified with OpenShift Container Platform 4.6 and NSX-T 3.x+.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See **Edit Time Configuration for a Host** in the VMware documentation.

11.7.5. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

11.7.5.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.
**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

**IMPORTANT**

All virtual machines must reside in the same datastore and in the same folder as the installer.

### 11.7.5.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in **initramfs** during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

### 11.7.5.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th>Machine</th>
<th>Operating System</th>
<th>vCPU</th>
<th>Virtual RAM</th>
<th>Storage</th>
<th>IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster
storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

11.7.5.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

11.7.6. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

Prerequisites

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

Procedure

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

11.7.6.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.
You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 11.49. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 11.50. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 11.51. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.
Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   Do not configure session persistence for an API load balancer.

   Configure the following ports on both the front and back of the load balancers:

   **Table 11.52. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

   **NOTE**
   The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.
A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

**Table 11.53. Application Ingress load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**Ethernet adaptor hardware address requirements**

When provisioning VMs for the cluster, the ethernet interfaces configured for each VM must use a MAC address from the VMware Organizationally Unique Identifier (OUI) allocation ranges:

- 00:05:69:00:00:00 to 00:05:69:FF:FF:FF
- 00:0c:29:00:00:00 to 00:0c:29:FF:FF:FF
- 00:1c:14:00:00:00 to 00:1c:14:FF:FF:FF
- 00:50:56:00:00:00 to 00:50:56:FF:FF:FF

If a MAC address outside the VMware OUI is used, the cluster installation will not succeed.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**Additional resources**

- Configuring chrony time service
11.7.6.2. User-provisioned DNS requirements

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

Table 11.54. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
Master hosts  
<master><n>.  
<cluster_name>.  
<base_domain>.  
DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.

Worker hosts  
<worker><n>.  
<cluster_name>.  
<base_domain>.  
Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 11.13. Sample DNS zone database**

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root (  
2019070700 ; serial 
3H ; refresh (3 hours)  
30M ; retry (30 minutes) 
2W ; expiry (2 weeks)  
1W ) ; minimum (1 week) 
IN NS ns1.example.com. 
IN MX 10 smtp.example.com.  
; 
ns1 IN A 192.168.1.5 
smt1p IN A 192.168.1.5  
; 
helper IN A 192.168.1.5 
helper.ocp4 IN A 192.168.1.5  
; 
; The api identifies the IP of your load balancer. 
api.ocp4 IN A 192.168.1.5 
api-int.ocp4 IN A 192.168.1.5  
; 
; The wildcard also identifies the load balancer. 
*.apps.ocp4 IN A 192.168.1.5  
; 
; Create an entry for the bootstrap host. 
bootstrap.ocp4 IN A 192.168.1.96  
; 
; Create entries for the master hosts. 
master0.ocp4 IN A 192.168.1.97 
master1.ocp4 IN A 192.168.1.98
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

```
$TTL 1W
@ IN SOA ns1.example.com. root (  
  2019070700 ; serial  
  3H ; refresh (3 hours)  
  30M ; retry (30 minutes)  
  2W ; expiry (2 weeks)  
  1W ) ; minimum (1 week)  
IN NS ns1.example.com.  
;  
; The syntax is "last octet" and the host must have an FQDN  
; with a trailing dot.  
97 IN PTR master0.ocp4.example.com.  
98 IN PTR master1.ocp4.example.com.  
99 IN PTR master2.ocp4.example.com.  
;  
96 IN PTR bootstrap.ocp4.example.com.  
;  
5 IN PTR api.ocp4.example.com.  
5 IN PTR api-int.ocp4.example.com.  
;  
11 IN PTR worker0.ocp4.example.com.  
7 IN PTR worker1.ocp4.example.com.  
;  
;EOF
```

Example 11.14. Sample DNS zone database for reverse records

11.7.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

```
NOTE
In a production environment, you require disaster recovery and debugging.
```

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.
NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

   ```
   $ ssh-add <path>/<file_name> 1
   ```

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa
Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

11.7.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

- Obtain the `imageContentSources` section from the output of the command to mirror the repository.

- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

   - Unless you use a registry that RHCOS trusts by default, such as `docker.io`, you must provide the contents of the certificate for your mirror repository in the `additionalTrustBundle` section. In most cases, you must provide the certificate for your mirror.

   - You must include the `imageContentSources` section from the output of the command to mirror the repository.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
11.7.8.1. Sample install-config.yaml file for VMware vSphere

You can customize the install-config.yaml file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
  controlPlane:
    hyperthreading: Enabled
    name: master
    replicas: 3
metadata:
  name: test
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
    datacenter: datacenter
    defaultDatastore: datastore
    folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"
fips: false
pullSecret: '{"auths":{"<local_registry>": {"auth": ";","email": ";"}}}'
sshKey: 'ssh-ed25519 AAAA...
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
    ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
    -----END CERTIFICATE-----
imageContentSources:
  - mirrors:
    - source: quay.io/openshift-release-dev/ocp-release
      source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2. The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, - , and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one
Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

You must set the value of the **replicas** parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.

The fully-qualified hostname or IP address of the vCenter server.

The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

The password associated with the vSphere user.

The vSphere datacenter.

The default vSphere datastore to use.

Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.
For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or

The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

Provide the contents of the certificate file that you used for your mirror registry.

Provide the `imageContentSources` section from the output of the command to mirror the repository.

11.7.8.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:@<ip>:<port>  
     httpsProxy: https://<username>:@<ip>:<port>  
     noProxy: example.com
   ```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

11.7.9. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.

- You created the install-config.yaml installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>  
   ```

   Example output

   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: install_dir/manifests and install_dir/openshift

   For `<installation_directory>`, specify the installation directory that contains the install-config.yaml file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```bash
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.

   - You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the mastersSchedulable parameter in the <installation_directory>/manifests/cluster-scheduler-02-config.yaml Kubernetes manifest file is set to false. This setting prevents pods from being scheduled on the control plane machines:
a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```
   $ ./openshift-install create ignition-configs --dir `<installation_directory>`
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   - auth
     - kubeadmin-password
     - kubeconfig
     - bootstrap.ign
     - master.ign
     - metadata.json
     - worker.ign

11.7.10. Configuring chrony time service

You must set the time server and related settings used by the chrony time service (`chronyd`) by modifying the contents of the `chrony.conf` file and passing those contents to your nodes as a machine config.

Procedure

1. Create the contents of the `chrony.conf` file and encode it as base64. For example:

   ```
   $ cat << EOF | base64
   pool 0.rhel.pool.ntp.org iburst
   driftfile /var/lib/chrony/drift
   makestep 1.0 3
   rttscync
   logdir /var/log/chrony
   EOF
   ```

   Specify any valid, reachable time source, such as the one provided by your DHCP server.

   Example output

   ```
   ICAGlHNlcnZlciBjBjG9jay5yZWRoYXQuY29tIjIiGloiAogICAgZHJpZnRmaWxlciBjIjIj
   L2Nocm9ueS9kcm9mdAo9IAgbWFriZXN0ZXAgMS4wlDMiICAgIHJ0Y3N5bmMiMiCAgIGxvZ2
   RpCiAv
   dmFyL2xvZy9jaHJvbnkK
   ```

2. Create the `MachineConfig` object file, replacing the base64 string with the one you just
created. This example adds the file to master nodes. You can change it to worker or make an additional MachineConfig for the worker role. Create MachineConfig files for each type of machine that your cluster uses:

```yaml
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
name: 99-masters-chrony-configuration
spec:
  config:
    ignition: {}
    security: {}
    timeouts: {}
    version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
        - contents:
            source: data:text/plain;charset=utf-8;base64,ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGliL2Nocm9ueS9kcmlmdAoAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmlMKnkK
            mode: 420
            overwrite: true
            path: /etc/chrony.conf
osImageURL: ""
EOF
```

1. Specify an octal value mode for the mode field in the machine config file. After creating the file and applying the changes, the mode is converted to a decimal value. You can check the YAML file with the command `oc get mc <mc-name> -o yaml`.

3. Make a backup copy of the configuration files.

4. Apply the configurations in one of two ways:
   - If the cluster is not up yet, after you generate manifest files, add this file to the `<installation_directory>/openshift` directory, and then continue to create the cluster.
   - If the cluster is already running, apply the file:
     ```
     $ oc apply -f ./99-masters-chrony-configuration.yaml
     ```

### 11.7.11. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware vSphere. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.
Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  ```
  $ jq -r .infraID <installation_directory>/metadata.json
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6
```

The output of this command is your cluster name and a random string.

11.7.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a vSphere cluster.

Procedure

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

```json
{
  "ignition": {
    "config": {
      "merge": [

      ]
    }
  }
}
```
Specify the URL of the bootstrap Ignition config file that you hosted.

When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

- `<installation_directory>/master.ign`
- `<installation_directory>/worker.ign`
- `<installation_directory>/merge-bootstrap.ign`

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

```bash
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.
The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   a. Click the **VMs and Templates** view.
   b. Right-click the name of your datacenter.
   c. Click **New Folder** → **New VM and Template Folder**.
   d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the `install-config.yaml` file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.

7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

   **NOTE**

   In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

   a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.
   b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.
   c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.
   d. On the **Select a compute resource** tab, click the name of your vSphere cluster.
   e. On the **Select storage** tab, configure the storage options for your VM.
      
      - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
      - Select the datastore that you specified in your `install-config.yaml` file.
   f. On the **Select network** tab, specify the network that you configured for the cluster, if available.
   g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

   **IMPORTANT**

   Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.
a. Right-click the template name and click **Clone → Clone to Virtual Machine**

b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as `control-plane-0` or `compute-1`.

c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.

d. On the **Select a compute resource** tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

e. Optional: On the **Select storage** tab, customize the storage options.

f. On the **Select clone options**, select **Customize this virtual machine’s hardware**

g. On the **Customize hardware** tab, click **VM Options → Advanced**.
   - Optional: Override default DHCP networking in vSphere. To enable static IP networking:
     - i. Set your static IP configuration:
       - $ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>:[iface]:none
         nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
       - **Example command**
         - $ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
           nameserver=8.8.8.8"
     - ii. Set the `guestinfo.afterburn.initrd.network-kargs` property before booting a VM from an OVA in vSphere:
       - $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
   - Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:
     - Memory reservation value must be equal to its configured memory size.
     - CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.
   - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
     - `guestinfo.ignition.config.data`: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
     - `guestinfo.ignition.config.data.encoding`: Specify `base64`.
     - `disk.EnableUUID`: Specify `TRUE`.

---

`CHAPTER 11. INSTALLING ON VSPHERE`
h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.

i. Complete the configuration and power on the VM.

9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

11.7.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

**Prerequisites**

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

**Procedure**

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click **Clone → Clone to Virtual Machine**
   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as compute-1.
   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
   d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
   e. Optional: On the **Select storage** tab, customize the storage options.
   f. On the **Select clone options**, select **Customize this virtual machine’s hardware**
   g. On the **Customize hardware** tab, click **VM Options → Advanced**.
      - From the **Latency Sensitivity** list, select **High**.
      - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
         - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
         - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
In the Virtual Hardware panel of the Customize hardware tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under Add network adapter if there are multiple networks available.

Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

11.7.14. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making /var or a subdirectory of /var, such as /var/lib/etcd, a separate partition, but not both.

**IMPORTANT**

Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to coreos-installer that allow you to retain existing data partitions.

Creating a separate /var partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the /var partition or a subdirectory of /var. For example:

- /var/lib/containers: Holds container-related content that can grow as more images and containers are added to a system.

- /var/lib/etcd: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.

- /var: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a /var directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.
Because */var* must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS),
the following procedure sets up the separate */var* partition by creating a machine config that is inserted
during the *openshift-install* preparation phases of an OpenShift Container Platform installation.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:
   
   ```
   $ mkdir $HOME/clusterconfig
   ```

2. Run *openshift-install* to create a set of files in the *manifest* and *openshift* subdirectories. Answer the system questions as you are prompted:
   
   ```
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadm-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...
   ```

3. Create a *MachineConfig* object and add it to a file in the *openshift* directory. For example, name the file *98-var-partition.yaml*, change the disk device name to the name of the storage device on the *worker* systems, and set the storage size as appropriate. This example places the */var* directory on a separate partition:

   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
     - machineconfiguration.openshift.io/role: worker
     name: 98-var-partition
   spec:
     config:
       ignition:
         version: 3.1.0
       storage:
         disks:
         - device: /dev/<device_name>  
           partitions:
             - label: var
               startMiB: <partition_start_offset>  
               sizeMiB: <partition_size>
           filesystems:
             - device: /dev/disk/by-partlabel/var
               path: /var
               format: xfs
         systemd:
           units:
             - name: var.mount
               enabled: true
               contents: |
               [Unit]
   ```
The storage device name of the disk that you want to partition.

When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

The size of the data partition in mebibytes.

The name of the mount unit must match the directory specified in the Where= directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named var-lib-containers.mount.

The prjquota mount option must be enabled for filesystems used for container storage.

NOTE
When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

11.7.15. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
Procedure

1. Monitor the bootstrap process:

   ```bash
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete
                   --log-level=info
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

```text
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   **IMPORTANT**

   You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

11.7.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:
11.7.17. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   $ oc get nodes

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>63m</td>
<td>v1.19.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>64m</td>
<td>v1.19.0</td>
</tr>
</tbody>
</table>

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   $ oc get csr

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-8b2br</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-8vmps</td>
<td>15m</td>
<td>system:serviceaccount:openshift-machine-config-operator:node-bootstrapper</td>
<td>Pending</td>
</tr>
</tbody>
</table>
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in Pending status, approve the CSRs for your cluster machines:

   **NOTE**

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

   **NOTE**

   For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, oc rsh, and oc logs commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the system:node or system:admin groups, and confirm the identity of the node.

   • To approve them individually, run the following command for each valid CSR:

   ```bash
   $ oc adm certificate approve <csr_name> 1
   ```

   1. **<csr_name>** is the name of a CSR from the list of current CSRs.

   • To approve all pending CSRs, run the following command:

   ```bash
   $ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' xargs --no-run-if-empty oc adm certificate approve
   ```

   **NOTE**

   Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

   ```bash
   $ oc get csr
   ```

   **Example output**
If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```shell
  $ oc adm certificate approve <csr_name>
  ``

  `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```shell
  $ oc adm certificate approve
  $ oc adm certificate approve
  $ oc adm certificate approve
  ``

After all client and server CSRs have been approved, the machines have the Ready status. Verify this by running the following command:

```shell
$ oc get nodes
```

### Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

### Additional information

For more information on CSRs, see [Certificate Signing Requests](#).

### 11.7.18. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

#### Prerequisites

- Your control plane has initialized.
Procedure

1. Watch the cluster components come online:

   ```
   $ watch -n5 oc get clusteroperators
   ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

### 11.7.18.1. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:
Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

11.7.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

11.7.18.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports ReadWriteOnce access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, ReadWriteMany access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.
Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**
   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```json
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```bash
   $ oc get clusteroperator image-registry
   ```

**11.7.18.2.2. Configuring storage for the image registry in non-production clusters**

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

  ```bash
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
  {"storage":{"emptyDir":{}}}}'
  ```
If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

### 11.7.18.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only **1** replica:

   ```
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
       - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   **1** A unique name that represents the **PersistentVolumeClaim** object.
The namespace for the PersistentVolumeClaim object, which is openshift-image-registry.

The access mode of the persistent volume claim. With ReadWriteOnce, the volume can be mounted with read and write permissions by a single node.

The size of the persistent volume claim.

b. Create the PersistentVolumeClaim object from the file:

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Edit the registry configuration so that it references the correct PVC:

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

**Example output**

```
storage:
  pvc:
    claim: 1
```

Creating a custom PVC allows you to leave the **claim** field blank for the default automatic creation of an **image-registry-storage** PVC.

For instructions about configuring registry storage so that it references the correct PVC, see Configuring the registry for vSphere.

11.7.19. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

```
$ watch -n5 oc get clusteroperators
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```
   $ oc get pods --all-namespaces
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2m</td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5m</td>
</tr>
</tbody>
</table>
   
   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```
   $ oc logs <pod_name> -n <namespace>
   ```

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Register your cluster on the Cluster registration page.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.
11.7.20. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

11.7.21. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

11.7.22. Next steps

- Customize your cluster.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can opt out of remote health reporting.

11.8. UNINSTALLING A CLUSTER ON VSPHERE THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE

You can remove a cluster that you deployed in your VMware vSphere instance by using installer-provisioned infrastructure.

NOTE

When you run the openshift-install destroy cluster command to uninstall OpenShift Container Platform, vSphere volumes are not automatically deleted. The cluster administrator must manually find the vSphere volumes and delete them.
11.8.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

**NOTE**

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

**Prerequisites**

- Have a copy of the installation program that you used to deploy the cluster.
- Have the files that the installation program generated when you created your cluster.

**Procedure**

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

   ```bash
   $ ./openshift-install destroy cluster \
   --dir <installation_directory> --log-level info
   ```

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different details, specify `warn`, `debug`, or `error` instead of `info`.

   **NOTE**

   You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.
CHAPTER 12. INSTALLING ON VMC

12.1. INSTALLING A CLUSTER ON VMC

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere by deploying it to VMware Cloud (VMC) on AWS.

Once you have configured your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

12.1.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
  - A DNS record for `api.<cluster_name>.<base_domain>` pointing to the allocated IP address.
  - A DNS record for `*.apps.<cluster_name>.<base_domain>` pointing to the allocated IP address.
- Configure the following firewall rules:
  - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the Internet. This is used by nodes and applications to download container images.
  - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
  - An HTTPS firewall rule between the OpenShift Container Platform compute network and
vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

- You must have the following information to deploy OpenShift Container Platform:
  - The OpenShift Container Platform cluster name, such as `vmc-prod-1`.
  - The base DNS name, such as `companyname.com`.
  - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to `10.128.0.0/14` and `172.30.0.0/16`, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
  - The following vCenter information:
    - vCenter hostname, username, and password
    - Datacenter name, such as `SDDC-Datacenter`
    - Cluster name, such as `Cluster-1`
    - Network name
    - Datastore name, such as `WorkloadDatastore`

  **NOTE**
  It is recommended to move your vSphere cluster to the VMC `Compute-ResourcePool` resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The `openshift-install` installation program
    - The OpenShift CLI (`oc`) tool

  **NOTE**
  You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.
  However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.1.1. VMC Sizer tool
VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
  - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

### 12.1.2. vSphere prerequisites

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

### 12.1.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.
IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

12.1.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

Table 12.1. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

IMPORTANT

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

12.1.5. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges
To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder
to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

### Example 12.1. Roles and privileges required for installation

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>Cns.Searchable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.AttachTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sessions.ValidateSession</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StorageProfile.View</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>Host.Config.Storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.AssignResourcePool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>--------------------</td>
</tr>
</tbody>
</table>
| Virtual Machine Folder        | Always        | Resource.AssignVMToPool  
VApp.Import  
VirtualMachine.Config.AddExistingDisk  
VirtualMachine.Config.AddNewDisk  
VirtualMachine.Config.AddRemoveDevice  
VirtualMachine.Config.AdvancedConfig  
VirtualMachine.Config.Annotation  
VirtualMachine.Config.CPUCount  
VirtualMachine.Config.DiskExtend  
VirtualMachine.Config.DiskLease  
VirtualMachine.Config.EditDevice  
VirtualMachine.Config.Memory  
VirtualMachine.Config.RemoveDisk  
VirtualMachine.Config.Rename  
VirtualMachine.Config.RestGuestInfo  
VirtualMachine.Config.Restore  
VirtualMachine.Config.Remove  
<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 12.2. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Installation program creates the folder</td>
<td>True</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.

Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.
Cluster resources
When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits
Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements
You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses
An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 12.2. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

12.1.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as `AWS key pairs`.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:
Specifying the SSH key:

1. Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   ```
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   **NOTE**

   Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**12.1.7. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**
- You have a computer that runs Linux or macOS, with 500 MB of local disk space

### Procedure

1. Access the **Infrastructure Provider** page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 12.1.8. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

#### Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.

2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

   ```
   certs
   ├── lin
   ```
Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```bash
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

Update your system trust. For example, on a Fedora operating system, run the following command:

```bash
# update-ca-trust extract
```

### 12.1.9. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the `create cluster` command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:

```bash
$ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

Provide values at the prompts:

a. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

b. Select `vsphere` as the platform to target.

c. Specify the name of your vCenter instance.

d. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.

The installation program connects to your vCenter instance.

e. Select the datacenter in your vCenter instance to connect to.

f. Select the default vCenter datastore to use.

g. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

h. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

i. Enter the virtual IP address that you configured for control plane API access.

j. Enter the virtual IP address that you configured for cluster ingress.

k. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

l. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

m. Paste the pull secret from the Red Hat OpenShift Cluster Manager.
IMPORTANT

Use the openshift-install command from the bastion hosted in the VMC environment.

NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the kubeadmin user, display in your terminal.

Example output

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

NOTE

The cluster access and credential information also outputs to <installation_directory>/openshift_install.log when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.
12.1.10. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

12.1.10.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ```

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:
   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the oc command:

```
$ oc <command>
```

12.1.10.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   To check your PATH, open the command prompt and execute the following command:
After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

### 12.1.10.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your PATH. To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   $ oc <command>
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   $ oc <command>
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

### 12.1.11. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
2. Verify you can run `oc` commands successfully using the exported configuration:

```
$ oc whoami
```

**Example output**

```
system:admin
```

### 12.1.12. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

#### 12.1.12.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from Removed to Managed.

**NOTE**

The Prometheus console provides an `ImageRegistryRemoved` alert, for example:

"Image Registry has been removed. `ImageStreamTags`, `BuildConfigs` and `DeploymentConfigs` which reference `ImageStreamTags` may not work as expected. Please configure storage and update the config to Managed state by editing
`configs.imageregistry.operator.openshift.io`.
"

#### 12.1.12.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

#### 12.1.12.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.
IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is **emptyDir**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.
4. Check the `clusteroperator` status:

   ```
   $ oc get clusteroperator image-registry
   ```

12.1.12.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only 1 replica:

   ```
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
   {"rolloutStrategy":"Recreate","replicas":1}}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere `PersistentVolumeClaim` object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
     namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   1. A unique name that represents the `PersistentVolumeClaim` object.
   2. The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.
   3. The access mode of the persistent volume claim. With `ReadWriteOnce`, the volume can be mounted with read and write permissions by a single node.
   4. The size of the persistent volume claim.

   b. Create the `PersistentVolumeClaim` object from the file:
$ oc create -f pvc.yaml -n openshift-image-registry

3. Edit the registry configuration so that it references the correct PVC:

$ oc edit config.imageregistry.operator.openshift.io -o yaml

Example output

```
storage:
pvc:
  claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see Configuring the registry for vSphere.

12.1.13. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

**Procedure**

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

12.1.14. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See About remote health monitoring for more information about the Telemetry service
12.1.15. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.2. INSTALLING A CLUSTER ON VMC WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance using installer-provisioned infrastructure by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

To customize the OpenShift Container Platform installation, you modify parameters in the `install-config.yaml` file before you install the cluster.

12.2.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
  - A DNS record for `api.<cluster_name>.<base_domain>` pointing to the allocated IP address.
  - A DNS record for `*.apps.<cluster_name>.<base_domain>` pointing to the allocated IP address.

- Configure the following firewall rules:
- An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the Internet. This is used by nodes and applications to download container images.

- An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.

- An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

You must have the following information to deploy OpenShift Container Platform:

- The OpenShift Container Platform cluster name, such as `vmc-prod-1`.
- The base DNS name, such as `companyname.com`.
- If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to `10.128.0.0/14` and `172.30.0.0/16`, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
- The following vCenter information:
  - vCenter hostname, username, and password
  - Datacenter name, such as `SDDC-Datacenter`
  - Cluster name, such as `Cluster-1`
  - Network name
  - Datastore name, such as `WorkloadDatastore`

  **NOTE**

  It is recommended to move your vSphere cluster to the VMC `Compute-ResourcePool` resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The `openshift-install` installation program
    - The OpenShift CLI (`oc`) tool
NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.2.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
  - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

12.2.2. vSphere prerequisites

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

12.2.3. Internet access for OpenShift Container Platform
In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 12.2.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 12.3. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.
12.2.5. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

**Required vCenter account privileges**

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

---

**Example 12.3. Roles and privileges required for installation**

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
| Virtual Machine Folder  | Always        | Resource.AssignVMToPool
<p>|                         |               | VApp.Import       |
|                         |               | VirtualMachine.Config.AddExistingDisk |
|                         |               | VirtualMachine.Config.AddNewDisk |
|                         |               | VirtualMachine.Config.AddRemoveDevice |
|                         |               | VirtualMachine.Config.AdvancedConfig |
|                         |               | VirtualMachine.Config.Annotation |
|                         |               | VirtualMachine.Config.CPUCount |
|                         |               | VirtualMachine.Config.DiskExtend |
|                         |               | VirtualMachine.Config.DiskLease |
|                         |               | VirtualMachine.Config&gt;EditDevice |
|                         |               | VirtualMachine.Config&gt;Memory |
|                         |               | VirtualMachine.Config&gt;RemoveDisk |
|                         |               | VirtualMachine.Config&gt;Rename |
|                         |               | VirtualMachine.Config&gt;RestartGuestInfo |
|                         |               | VirtualMachine.Config&gt;Resource |
|                         |               | VirtualMachine.Config&gt;Settings |
|                         |               | VirtualMachine.Config&gt;UpgradeVirtualHardware |
|                         |               | VirtualMachine.Interact.GuestControl |
|                         |               | VirtualMachine.Interact.PowerOff |
|                         |               | VirtualMachine.Interact.PowerOn |
|                         |               | VirtualMachine.Interact&gt;Reset |
|                         |               | VirtualMachine.Inventory&gt;Create |
|                         |               | VirtualMachine.Inventory&gt;CreateFromExisting |
|                         |               | VirtualMachine.Inventory&gt;Delete |
|                         |               | VirtualMachine.Provisioning.Clone |</p>
<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>If the installation program creates the virtual machine folder</td>
<td>Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddExistingDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddRemoveDevice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AdvancedConfig</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Annotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.CPUCount</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.DiskExtend</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.DiskLease</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.EditDevice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Memory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RemoveDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Rename</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RestGuestInfo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RestoreSettings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UpgradeVirtualHardware</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.GuestControl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOff</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.Rest</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Create</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.CreateFromExisting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Provisioning.Clone</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Folder.Create</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Folder.Delete</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

### Example 12.4. Required permissions and propagation settings

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Installation program creates the folder</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see [vSphere Permissions and User Management Tasks](#) in the vSphere documentation.

### Using OpenShift Container Platform with vMotion

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.
Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

**Cluster resources**

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- **Virtual machines:**
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

**Cluster limits**

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

**Networking requirements**

You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

**Required IP Addresses**

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.
You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

12.2.6. Generating an SSH private key and adding it to the agent
If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**
In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**
You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```bash
$ ssh-keygen -t ed25519 -N "\n-f <path>/<file_name>
```

Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

```bash
$ eval "$(ssh-agent -s)"
```

**Example output**

Agent pid 31874

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```bash
$ ssh-add <path>/<file_name>
```

**Example output**

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**12.2.7. Obtaining the installation program**
Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

**Procedure**

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

**IMPORTANT**

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

**IMPORTANT**

Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar xvf openshift-install-linux.tar.gz
```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

**12.2.8. Adding vCenter root CA certificates to your system trust**

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

**Procedure**

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crt
    ├── 7e757f6a.0.crt
    └── 8e4f8471.0.crt
```

3 directories, 15 files

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

12.2.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory>
   ```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select `vsphere` as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.
   The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.
2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

**IMPORTANT**

The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

**12.2.9.1. Installation configuration parameters**

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

**12.2.9.1.1. Required configuration parameters**

Required installation configuration parameters are described in the following table:

**Table 12.5. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>baseDomain</strong></td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>&lt;metadata.name&gt;</code> and <code>&lt;baseDomain&gt;</code> values that uses the <code>&lt;metadata.name&gt;..&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td><strong>metadata</strong></td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td><strong>metadata.name</strong></td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td><strong>platform</strong></td>
<td>The configuration for the specific platform upon which to perform the installation: <code>aws</code>, <code>baremetal</code>, <code>azure</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>. For additional information about <code>platform.&lt;platform&gt;</code> parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
12.2.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 12.6. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
<tr>
<td>networking.network</td>
<td>The cluster network provider Container Network Interface (CNI) plug-in to install.</td>
<td>Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.</td>
</tr>
<tr>
<td>networking.clusterNetwork</td>
<td>The IP address blocks for pods.</td>
<td>An array of objects. For example:</td>
</tr>
</tbody>
</table>
|                   | The default value is 10.128.0.0/14 with a host prefix of /23. | networking:
|                   | If you specify multiple IP address blocks, the blocks must not overlap. |   clusterNetwork:
|                   |                                                          |     - cidr: 10.128.0.0/14
<p>|                   |                                                          |       hostPrefix: 23                                                 |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use <code>networking.clusterNetwork</code>. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if <code>hostPrefix</code> is set to 23 then each node is assigned a /23 subnet out of the given <code>cidr</code>. A <code>hostPrefix</code> value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
</tbody>
</table>
| networking.serviceNetwork | The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network. | An array with an IP address block in CIDR format. For example: ```json
networking:
  serviceNetwork:
  - 172.30.0.0/16
``` |
| networking.machineNetwork | The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap. | An array of objects. For example: ```json
networking:
  machineNetwork:
  - cidr: 10.0.0.0/16
``` |
| networking.machineNetwork.cidr | Required if you use `networking.machineNetwork`. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24. | An IP network block in CIDR notation. For example, 10.0.0.0/16. **NOTE** Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in. |

### 12.2.9.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Table 12.7. Optional parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <strong>hyperthreading</strong>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
</tr>
<tr>
<td></td>
<td>IMPORTANT</td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hypertreading</em>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use <code>controlPlane</code>. The name of the machine pool.</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use <code>controlPlane</code>. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <code>compute.platform</code> parameter value.</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
</tr>
</tbody>
</table>
The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.

### NOTE

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
</tbody>
</table>

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
### publish

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

*Internal* or *External*. The default value is *External*.

Setting this field to *Internal* is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to *Internal*, the cluster will become non-functional. For more information, refer to [BZ#1953035](#).

### sshKey

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

---

### 12.2.9.14. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.vCenter</code></td>
<td>The fully-qualified hostname or IP address of the vCenter server.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.username</code></td>
<td>The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.password</code></td>
<td>The password for the vCenter user name.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.datacenter</code></td>
<td>The name of the datacenter to use in the vCenter instance.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.defaultDatastore</code></td>
<td>The name of the default datastore to use for provisioning volumes.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.folder</code></td>
<td>Optional. The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the datacenter virtual machine folder.</td>
<td>String, for example, <code>/&lt;datacenter_name&gt;/vm/&lt;folder_name&gt;/&lt;subfolder_name&gt;</code>.</td>
</tr>
<tr>
<td><code>platform.vsphere.network</code></td>
<td>The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.cluster</code></td>
<td>The vCenter cluster to install the OpenShift Container Platform cluster in.</td>
<td>String</td>
</tr>
<tr>
<td><code>platform.vsphere.apiVIP</code></td>
<td>The virtual IP (VIP) address that you configured for control plane API access.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td><code>platform.vsphere.ingressVIP</code></td>
<td>The virtual IP (VIP) address that you configured for cluster ingress.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
</tbody>
</table>

### 12.2.9.15. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.clusterOSImage</code></td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <a href="https://mirror.openshift.com/images/rhcos-">https://mirror.openshift.com/images/rhcos-</a>&lt;version&gt;-vmware.&lt;architecture&gt;.ova.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>platform.vsphere.os</td>
<td>The size of the disk in gigabytes.</td>
<td>Integer</td>
</tr>
<tr>
<td>Disk.diskSizeGB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>platform.vsphere.cpus</td>
<td>The total number of virtual processor cores to assign a virtual machine.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.coresPerSocket</td>
<td>The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is platform.vsphere.cpus/platform.vsphere.coresPerSocket. The default value is 1.</td>
<td>Integer</td>
</tr>
<tr>
<td>platform.vsphere.memoryMB</td>
<td>The size of a virtual machine’s memory in megabytes.</td>
<td>Integer</td>
</tr>
</tbody>
</table>

### 12.2.9.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com  # 1
compute:  # 2
  - name: worker
    replicas: 3
    platform:
      vsphere:  # 4
        cpus: 2
        coresPerSocket: 2
        memoryMB: 8196
        osDisk:
          diskSizeGB: 120
controlPlane:  # 5
  name: master
  replicas: 3
  platform:
    vsphere:  # 7
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster  # 8
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, `-`, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.

The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

### 12.2.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s spec.noProxy field to bypass the proxy if necessary.

**NOTE**

The Proxy object status.noProxy field is populated with the values of the networking.machineNetwork[].cidr, networking.clusterNetwork[].cidr, and networking.serviceNetwork[] fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object status.noProxy field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your install-config.yaml file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
   noProxy: example.com 3
   additionalTrustBundle: | 4
      -----BEGIN CERTIFICATE-----
      <MY_TRUSTED_CA_CERT>
      -----END CERTIFICATE-----
      ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.
   4 If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.
NOTE
The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

NOTE
Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

12.2.10. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT
You can run the `create cluster` command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
   ```

   1 For `<installation_directory>`, specify the location of your customized `install-config.yaml` file.
   2 To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   IMPORTANT
   Use the `openshift-install` command from the bastion hosted in the VMC environment.
NOTE

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the kubeadmin user, display in your terminal.

Example output

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
INFO Time elapsed: 36m22s
```

NOTE

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

IMPORTANT

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

12.2.11. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.
IMPORTANT

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

12.2.11.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH. To check your PATH, execute the following command:

   $ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

   $ oc <command>

12.2.11.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH. To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:
12.2.11.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the oc binary to a directory on your PATH.
   To check your PATH, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the oc command:

   ```
   $ oc <command>
   ```

12.2.12. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```
12.2.13. Creating registry storage

After you install the cluster, you must create storage for the Registry Operator.

12.2.13.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

**NOTE**

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

12.2.13.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

12.2.13.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.
IMPORTANT

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   
   **NOTE**

   If the storage type is **emptyDir**, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   
   **Example output**

   ```json
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an **image-registry-storage** PVC.
4. Check the `clusteroperator` status:

```
$ oc get clusteroperator image-registry
```

### 12.2.13.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only **1** replica:

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec": {"rolloutStrategy": "Recreate","replicas":1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere `PersistentVolumeClaim` object:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage
  namespace: openshift-image-registry
spec:
  accessModes:
    -ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

   1 A unique name that represents the `PersistentVolumeClaim` object.
   2 The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.
   3 The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
   4 The size of the persistent volume claim.

   b. Create the `PersistentVolumeClaim` object from the file:
3. Edit the registry configuration so that it references the correct PVC:

```bash
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

**Example output**

```
storage:
pvc:
  claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

### 12.2.14. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See [Snapshot Limitations](#) for more information.

**Procedure**

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

### 12.2.15. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to [OpenShift Cluster Manager](#).

After you confirm that your [OpenShift Cluster Manager](#) inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

**Additional resources**

- See [About remote health monitoring](#) for more information about the Telemetry service
12.2.16. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.3. INSTALLING A CLUSTER ON VMC WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance using installer-provisioned infrastructure with customized network configuration options by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

By customizing your OpenShift Container Platform network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing VXLAN configurations. To customize the installation, you modify parameters in the `install-config.yaml` file before you install the cluster. You must set most of the network configuration parameters during installation, and you can modify only `kubeProxy` configuration parameters in a running cluster.

12.3.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
  - A DNS record for `api.<cluster_name>.<base_domain>` pointing to the allocated IP address.
- A DNS record for *.apps.<cluster_name>.<base_domain> pointing to the allocated IP address.

- Configure the following firewall rules:
  - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the Internet. This is used by nodes and applications to download container images.
  - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
  - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

- You must have the following information to deploy OpenShift Container Platform:
  - The OpenShift Container Platform cluster name, such as vmc-prod-1.
  - The base DNS name, such as companyname.com.
  - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to 10.128.0.0/14 and 172.30.0.0/16, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
  - The following vCenter information:
    - vCenter hostname, username, and password
    - Datacenter name, such as SDDC-Datacenter
    - Cluster name, such as Cluster-1
    - Network name
    - Datastore name, such as WorkloadDatastore

  **NOTE**
  It is recommended to move your vSphere cluster to the VMC Compute-ResourcePool resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The `openshift-install` installation program
    - The OpenShift CLI (`oc`) tool
NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.3.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
  - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

12.3.2. vSphere prerequisites

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

12.3.3. Internet access for OpenShift Container Platform
In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access **Quay.io** to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 12.3.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 12.10. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See **Edit Time Configuration for a Host** in the VMware documentation.
### 12.3.5. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

#### Required vCenter account privileges

To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

#### Example 12.5. Roles and privileges required for installation

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>Cns.Searchable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.AttachTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sessions.ValidateSession View</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StorageProfile.View</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>Host.Config.Storage Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.AssignResourcePool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Add NewDisk</td>
</tr>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Add NewDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Add RemoveDevice</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AdvancedConfig</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Annotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.CPU Count</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Disk Extend</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Disk Lease</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Edit Device</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Memory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RemoveDisk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Rename</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.RestGuestInfo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.Settings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.UpgradeVirtualHardware</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.GuestControl</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOff</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.PowerOn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Interact.Reset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Create</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.CreateFromExisting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Inventory.Delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Provisioning.Clone</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

**Example 12.6. Required permissions and propagation settings**

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>Datacenter</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td></td>
<td></td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td></td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td></td>
<td></td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td></td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td></td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td></td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td></td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td></td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see *vSphere Permissions and User Management Tasks* in the vSphere documentation.

**Using OpenShift Container Platform with vMotion**

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.
Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

**Cluster resources**

When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- **Virtual machines:**
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

**Cluster limits**

Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

**Networking requirements**

You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

**Required IP Addresses**

An installer-provisioned vSphere installation requires two static IP addresses:

- The **API** address is used to access the cluster API.
- The **Ingress** address is used for cluster ingress traffic.
You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

**DNS records**
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

### Table 12.11. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

### 12.3.6. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**
1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" \
   -f <path>/<file_name> 1
```

1 Specify the path and file name, such as ~/.ssh/id_rsa, of the new SSH key. If you have an existing key pair, ensure your public key is in the your ~/.ssh directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the ssh-agent process as a background task:

```
$ eval "$(ssh-agent -s)"
```

**Example output**

```
Agent pid 31874
```

**NOTE**

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the ssh-agent:

```
$ ssh-add <path>/<file_name> 1
```

**Example output**

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 Specify the path and file name for your SSH private key, such as ~/.ssh/id_rsa

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

**12.3.7. Obtaining the installation program**
Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

12.3.8. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click Download trusted root CA certificates in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.
2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    └── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl
```

3 directories, 15 files

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

```
# update-ca-trust extract
```

### 12.3.9. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Create the `install-config.yaml` file.
   a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory>
```
For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

**IMPORTANT**

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

ii. Select vsphere as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster.

The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.
2. Modify the `install-config.yaml` file. You can find more information about the available parameters in the Installation configuration parameters section.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**
   
   The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 12.3.9.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 12.3.9.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 12.12. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is <code>v1</code>. The installer may also support older API versions.</td>
<td>String</td>
</tr>
</tbody>
</table>
The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the <metadata.name>. <baseDomain> format.

A fully-qualified domain or subdomain name, such as example.com.

Kubernetes resource ObjectMeta, from which only the name parameter is consumed.

The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.

String of lowercase letters, hyphens (-), and periods (.), such as dev.

The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.<platform> parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the baseDomain and metadata.name parameter values that uses the &lt;metadata.name&gt;. &lt;baseDomain&gt; format.</td>
<td>A fully-qualified domain or subdomain name, such as example.com.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource ObjectMeta, from which only the name parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of {{.metadata.name}}. {{.baseDomain}}.</td>
<td>String of lowercase letters, hyphens (-), and periods (.), such as dev.</td>
</tr>
<tr>
<td>platform</td>
<td>The configuration for the specific platform upon which to perform the installation: aws, baremetal, azure, openstack, ovirt, vsphere. For additional information about platform.&lt;platform&gt; parameters, consult the following table for your specific platform.</td>
<td>Object</td>
</tr>
</tbody>
</table>
12.3.9.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 12.13. Network parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>pullSecret</td>
<td>Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.</td>
<td></td>
</tr>
<tr>
<td>networking.network</td>
<td>The configuration for the cluster network.</td>
<td>Object</td>
</tr>
</tbody>
</table>

NOTE

You cannot modify parameters specified by the networking object after installation.

networking.network
The cluster network provider Container Network Interface (CNI) plug-in to install.

Either OpenShiftSDN or OVNKubernetes. The default value is OpenShiftSDN.

networking.clusterNetwork
The IP address blocks for pods.

The default value is 10.128.0.0/14 with a host prefix of /23.

If you specify multiple IP address blocks, the blocks must not overlap.

An array of objects. For example:

```
{  
  "networking": {  
    "clusterNetwork": {  
      "cidr": "10.128.0.0/14",  
      "hostPrefix": 23
    }
  }
}
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>networking.clusterNetwork.cidr</td>
<td>Required if you use networking.clusterNetwork. An IP address block. An IPv4 network.</td>
<td>An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between 0 and 32.</td>
</tr>
<tr>
<td>networking.clusterNetwork.hostPrefix</td>
<td>The subnet prefix length to assign to each individual node. For example, if hostPrefix is set to 23 then each node is assigned a /23 subnet out of the given cidr. A hostPrefix value of 23 provides 510 (2^(32 - 23) - 2) pod IP addresses.</td>
<td>A subnet prefix. The default value is 23.</td>
</tr>
<tr>
<td>networking.serviceNetwork</td>
<td>The IP address block for services. The default value is 172.30.0.0/16. The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.</td>
<td>An array with an IP address block in CIDR format. For example: networking: serviceNetwork: - 172.30.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork</td>
<td>The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.</td>
<td>An array of objects. For example: networking: machineNetwork: - cidr: 10.0.0.0/16</td>
</tr>
<tr>
<td>networking.machineNetwork.cidr</td>
<td>Required if you use networking.machineNetwork. An IP address block. The default value is 10.0.0.0/16 for all platforms other than libvirt. For libvirt, the default value is 192.168.126.0/24.</td>
<td>An IP network block in CIDR notation. For example, 10.0.0.0/16.</td>
</tr>
</tbody>
</table>

**NOTE**
Set the networking.machineNetwork to match the CIDR that the preferred NIC resides in.

### 12.3.9.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 12.14. Optional parameters**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes’ trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>compute.hyperthreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>compute.name</td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td>worker</td>
</tr>
<tr>
<td>compute.platform</td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws</code>, <code>azure</code>, <code>gcp</code>, <code>openstack</code>, <code>ovirt</code>, <code>vsphere</code>, or <code>{}</code></td>
</tr>
<tr>
<td>compute.replicas</td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to <code>2</code>. The default value is <code>3</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>controlPlane</td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <em>MachinePool</em> objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>controlPlane.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <em>amd64</em> (the default).</td>
<td>String</td>
</tr>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hypertreading</em>, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the <em>compute.platform</em> parameter value.</td>
<td><em>aws</em>, <em>azure</em>, <em>gcp</em>, <em>openstack</em>, <em>ovirt</em>, <em>vsphere</em>, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>fips</td>
<td>Enable or disable FIPS mode. The default is <strong>false</strong> (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
</tbody>
</table>

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageContentSources</td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td>imageContentSources.source</td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td>imageContentSources.mirrors</td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

Internal or External. The default value is External.

Setting this field to Internal is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to Internal, the cluster will become non-functional. For more information, refer to BZ#1953035.

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```
sshKey:
  <key1>
  <key2>
  <key3>
```

### 12.3.9.1.4. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table:

**Table 12.15. Additional VMware vSphere cluster parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.vCenter</td>
<td>The fully-qualified hostname or IP address of the vCenter server.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.username</td>
<td>The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter Description Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.password</td>
<td>The password for the vCenter user name.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.datacenter</td>
<td>The name of the datacenter to use in the vCenter instance.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.defaultDatastore</td>
<td>The name of the default datastore to use for provisioning volumes.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.folder</td>
<td><em>Optional.</em> The absolute path of an existing folder where the installation program creates the virtual machines. If you do not provide this value, the installation program creates a folder that is named with the infrastructure ID in the datacenter virtual machine folder.</td>
<td>String, for example, /&lt;datacenter_name&gt;/vm/&lt;folder_name&gt;/&lt;subfolder_name&gt;.</td>
</tr>
<tr>
<td>platform.vsphere.network</td>
<td>The network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.cluster</td>
<td>The vCenter cluster to install the OpenShift Container Platform cluster in.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.apiVIP</td>
<td>The virtual IP (VIP) address that you configured for control plane API access.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
<tr>
<td>platform.vsphere.ingressVIP</td>
<td>The virtual IP (VIP) address that you configured for cluster ingress.</td>
<td>An IP address, for example 128.0.0.1.</td>
</tr>
</tbody>
</table>

### 12.3.9.15. Optional VMware vSphere machine pool configuration parameters

Optional VMware vSphere machine pool configuration parameters are described in the following table:

#### Table 12.16. Optional VMware vSphere machine pool parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.clusterOSImage</td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <a href="https://mirror.openshift.com/images/rhcos-">https://mirror.openshift.com/images/rhcos-</a>&lt;version&gt;-vmware.&lt;architecture&gt;.ova.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td><code>platform.vsphere.osDisk.diskSizeGB</code></td>
<td>The size of the disk in gigabytes.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.cpus</code></td>
<td>The total number of virtual processor cores to assign a virtual machine.</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.coresPerSocket</code></td>
<td>The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is <code>platform.vsphere.cpus/platform.vsphere.coresPerSocket</code>. The default value is 1</td>
<td>Integer</td>
</tr>
<tr>
<td><code>platform.vsphere.memoryMB</code></td>
<td>The size of a virtual machine’s memory in megabytes.</td>
<td>Integer</td>
</tr>
</tbody>
</table>

12.3.9.2. Sample `install-config.yaml` file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 3
    platform:
      vsphere:
        cpus: 2
        coresPerSocket: 2
        memoryMB: 8196
        osDisk:
          diskSizeGB: 120
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
  platform:
    vsphere:
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, `-`, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

---

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.

The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.
12.3.9.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> 1
     httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
     noProxy: example.com 3
     additionalTrustBundle: |
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
   ...
   ```

   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

   2 A proxy URL to use for creating HTTPS connections outside the cluster.

   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to
additionalTrustBundle and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster` `Proxy` object is still created, but it will have a nil `spec`.

NOTE

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

12.3.10. Network configuration phases

When specifying a cluster configuration prior to installation, there are several phases in the installation procedures when you can modify the network configuration:

**Phase 1**

After entering the `openshift-install create install-config` command. In the `install-config.yaml` file, you can customize the following network-related fields:

- `networking.networkType`
- `networking.clusterNetwork`
- `networking.serviceNetwork`
- `networking.machineNetwork`

For more information on these fields, refer to "Installation configuration parameters".

NOTE

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

**Phase 2**

After entering the `openshift-install create manifests` command. If you must specify advanced network configuration, during this phase you can define a customized Cluster Network Operator manifest with only the fields you want to modify.

You cannot override the values specified in phase 1 in the `install-config.yaml` file during phase 2. However, you can further customize the cluster network provider during phase 2.
12.3.11. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- Create the `install-config.yaml` file and complete any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`

   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```bash
   $ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   EOF
   ```

   where:

   `<installation_directory>`

   Specifies the directory name that contains the `manifests/` directory for your cluster.

3. Open the `cluster-network-03-config.yml` file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

   **Specify a different VXLAN port for the OpenShift SDN network provider**

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: Network
   metadata:
     name: cluster
   spec:
   ```
4. Save the `cluster-network-03-config.yml` file and quit the text editor.

5. Optional: Back up the `manifests/cluster-network-03-config.yml` file. The installation program deletes the `manifests/` directory when creating the cluster.

### 12.3.12. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the `Network` API in the `operator.openshift.io` API group.

The CNO configuration inherits the following fields during cluster installation from the `Network` API in the `Network.config.openshift.io` API group and these fields cannot be changed:

- **clusterNetwork**: IP address pools from which pod IP addresses are allocated.
- **serviceNetwork**: IP address pool for services.
- **defaultNetwork.type**: Cluster network provider, such as OpenShift SDN or OVN-Kubernetes.

You can specify the cluster network provider configuration for your cluster by setting the fields for the `defaultNetwork` object in the CNO object named `cluster`.

#### 12.3.12.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always <code>cluster</code>.</td>
</tr>
<tr>
<td>spec.clusterNetwork</td>
<td>array</td>
<td>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</td>
</tr>
</tbody>
</table>
|                         |          | ``` specs:  
|                         |          |   - clusterNetwork:  
|                         |          |     - cidr: 10.128.0.0/19  
|                         |          |     hostPrefix: 23  
|                         |          |     - cidr: 10.128.32.0/19  
|                         |          |     hostPrefix: 23  
|                         |          | This value is ready-only and specified in the `install-config.yaml` file.                                                                 |
| defaultNetwork         |          |                                                                                                                                              |
A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:

```
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.

defaultNetwork object configuration
The values for the `defaultNetwork` object are defined in the following table:

**Table 12.18. defaultNetwork object**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>type</code></td>
<td>string</td>
<td>Either <strong>OpenShiftSDN</strong> or <strong>OVN Kubernetes</strong>. The cluster network provider is selected during installation. This value cannot be changed after cluster installation. <strong>NOTE</strong> OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.</td>
</tr>
<tr>
<td><code>openshiftSDNConfig</code></td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td><code>ovnKubernetesConfig</code></td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

Configuration for the OpenShift SDN CNI cluster network provider
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

```yaml
spec:
  serviceNetwork:
    - 172.30.0.0/14
```

This value is ready-only and specified in the `install-config.yaml` file.
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| mode        | string | Configures the network isolation mode for OpenShift SDN. The default value is **NetworkPolicy**.  
The values **Multitenant** and **Subnet** are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation. |
| mtu         | integer| The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.  
If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.  
If your cluster requires different MTU values for different nodes, you must set this value to 50 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1450.  
This value cannot be changed after cluster installation. |
| vxlanPort   | integer| The port to use for all VXLAN packets. The default value is 4789. This value cannot be changed after cluster installation.  
If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.  
On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port 9000 and port 9999. |

Example OpenShift SDN configuration

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

Configuration for the OVN-Kubernetes CNI cluster network provider

To configure the OVN-Kubernetes CNI cluster network provider:
The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

Table 12.20. ovnKubernetesConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to (100) less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of (9001), and some have an MTU of (1500), you must set this value to (1400). This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td>genevePort</td>
<td>integer</td>
<td>The port to use for all Geneve packets. The default value is (6081). This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>

Example OVN-Kubernetes configuration

```yaml
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

kubeProxyConfig object configuration

The values for the `kubeProxyConfig` object are defined in the following table:

Table 12.21. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>


### iptablesSyncPeriod

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>iptablesSyncPeriod</strong></td>
<td><strong>string</strong></td>
<td>The refresh period for <strong>iptables</strong> rules. The default value is <strong>30s</strong>. Valid suffixes include <strong>s</strong>, <strong>m</strong>, and <strong>h</strong> and are described in the <strong>Go time package documentation</strong>.</td>
</tr>
</tbody>
</table>

**NOTE**

Because of performance improvements introduced in OpenShift Container Platform 4.3 and greater, adjusting the **iptablesSyncPeriod** parameter is no longer necessary.

### proxyArguments.iptables-min-sync-period

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>proxyArguments.iptables-min-sync-period</strong></td>
<td><strong>array</strong></td>
<td>The minimum duration before refreshing <strong>iptables</strong> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include <strong>s</strong>, <strong>m</strong>, and <strong>h</strong> and are described in the <strong>Go time package</strong>. The default value is:</td>
</tr>
</tbody>
</table>

```
kubeProxyConfig:
  proxyArguments:
    iptables-min-sync-period:
      - 0s
```

### 12.3.13. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

**IMPORTANT**

You can run the **create cluster** command of the installation program only once, during initial installation.

**Prerequisites**

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and initialize the cluster deployment:
For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

**IMPORTANT**

Use the `openshift-install` command from the bastion hosted in the VMC environment.

**NOTE**

If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

**Example output**

```
... INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wt5AL"
INFO Time elapsed: 36m22s
```

**NOTE**

The cluster access and credential information also outputs to `<installation_directory>/.openshift_install.log` when an installation succeeds.
The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

12.3.14. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

12.3.14.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   $ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   $ echo $PATH
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

12.3.14.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the `oc` binary to a directory that is on your `PATH`. To check your `PATH`, open the command prompt and execute the following command:

```
C:\> path
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

12.3.14.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**

2. Select the appropriate version in the Version drop-down menu.
3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.
5. Move the `oc` binary to a directory on your `PATH`. To check your `PATH`, open a terminal and execute the following command:

```
$ echo $PATH
```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```
12.3.15. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure

1. Export the kubeadmin credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   
   For <installation_directory>, specify the path to the directory that you stored the installation files in.
   ```

2. Verify you can run oc commands successfully using the exported configuration:

   ```
   $ oc whoami
   
   Example output
   
   system:admin
   ```

12.3.16. Creating registry storage

After you install the cluster, you must create storage for the registry Operator.

12.3.16.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

1986
12.3.16.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

12.3.16.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports `ReadWriteOnce` access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, `ReadWriteMany` access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

**NOTE**

When using shared storage, review your security settings to prevent outside access.
2. Verify that you do not have a registry pod:

```bash
$ oc get pod -n openshift-image-registry
```

**NOTE**

If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

```bash
$ oc edit configs.imageregistry.operator.openshift.io
```

**Example output**

```yaml
storage:
pvc:
  claim: 1

1 Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.
```

4. Check the `clusteroperator` status:

```bash
$ oc get clusteroperator image-registry
```

12.3.16.2.2. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the `Recreate` rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only 1 replica:

```bash
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
  "rolloutStrategy":"Recreate","replicas":1}'}
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere PersistentVolumeClaim object:

```yaml
kind: PersistentVolumeClaim
```
A unique name that represents the *PersistentVolumeClaim* object.

The namespace for the *PersistentVolumeClaim* object, which is `openshift-image-registry`.

The access mode of the persistent volume claim. With *ReadWriteOnce*, the volume can be mounted with read and write permissions by a single node.

The size of the persistent volume claim.

b. Create the *PersistentVolumeClaim* object from the file:

```bash
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Edit the registry configuration so that it references the correct PVC:

```bash
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

**Example output**

```
storage:
pvc:
  claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an *image-registry-storage* PVC.

For instructions about configuring registry storage so that it references the correct PVC, see Configuring the registry for vSphere.

### 12.3.17. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

**Procedure**

To create a backup of persistent volumes:
1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

12.3.18. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources
- See About remote health monitoring for more information about the Telemetry service

12.3.19. Next steps
- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.4. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure in a restricted network by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

12.4.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.
You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Allocate two IP addresses, outside the DHCP range, and configure them with reverse DNS records.
  - A DNS record for `api.<cluster_name>.<base_domain>` pointing to the allocated IP address.
  - A DNS record for `*.apps.<cluster_name>.<base_domain>` pointing to the allocated IP address.

- Configure the following firewall rules:
  - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
  - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

- You must have the following information to deploy OpenShift Container Platform:
  - The OpenShift Container Platform cluster name, such as `vmc-prod-1`.
  - The base DNS name, such as `companyname.com`.
  - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to `10.128.0.0/14` and `172.30.0.0/16`, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
  - The following vCenter information:
    - vCenter hostname, username, and password
    - Datacenter name, such as `SDDC-Datacenter`
    - Cluster name, such as `Cluster-1`
Network name

Datastore name, such as WorkloadDatastore

**NOTE**

It is recommended to move your vSphere cluster to the VMC Compute-ResourcePool resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The openshift-install installation program
    - The OpenShift CLI (oc) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.4.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

12.4.2. vSphere prerequisites

- Create a registry on your mirror host and obtain the `imageContentSources` data for your version of OpenShift Container Platform.

  **IMPORTANT**
  
  Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

  **NOTE**
  
  If you are configuring a proxy, be sure to also review this site list.

12.4.3. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

**12.4.3.1. Additional limits**

Clusters in restricted networks have the following additional limitations and restrictions:

- The `ClusterVersion` status includes an **Unable to retrieve available updates** error.

- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

12.4.4. Internet access for OpenShift Container Platform
In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access **Quay.io** to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 12.4.5. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 12.22. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See **Edit Time Configuration for a Host** in the VMware documentation.
12.4.6. vCenter requirements

Before you install an OpenShift Container Platform cluster on your vCenter that uses infrastructure that the installer provisions, you must prepare your environment.

Required vCenter account privileges
To install an OpenShift Container Platform cluster in a vCenter, the installation program requires access to an account with privileges to read and create the required resources. Using an account that has global administrative privileges is the simplest way to access all of the necessary permissions.

If you cannot use an account with global administrative privileges, you must create roles to grant the privileges necessary for OpenShift Container Platform cluster installation. While most of the privileges are always required, some are required only if you plan for the installation program to provision a folder to contain the OpenShift Container Platform cluster on your vCenter instance, which is the default behavior. You must create or amend vSphere roles for the specified objects to grant the required privileges.

An additional role is required if the installation program is to create a vSphere virtual machine folder.

Example 12.7. Roles and privileges required for installation

<table>
<thead>
<tr>
<th>vSphere object for role</th>
<th>When required</th>
<th>Required privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>Cns.Searchable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.AttachTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.CreateTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.DeleteTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditCategory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InventoryService.Tagging.EditTag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sessions.ValidateSession</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StorageProfile.View</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>Host.Config.Storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource.AssignVMToPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.AssignResourcePool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VApp.Import</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VirtualMachine.Config.AddNewDisk</td>
</tr>
<tr>
<td>vSphere Datastore</td>
<td>Always</td>
<td>Datastore.AllocateSpace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.Browse</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Datastore.FileManagement</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>Network.Assign</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>vSphere object for role</td>
<td>When required</td>
<td>Required privileges</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Additionally, the user requires some **ReadOnly** permissions, and some of the roles require permission to propagate the permissions to child objects. These settings vary depending on whether or not you install the cluster into an existing folder.

**Example 12.8. Required permissions and propagation settings**

<table>
<thead>
<tr>
<th>vSphere object</th>
<th>Folder type</th>
<th>Propagate to children</th>
<th>Permissions required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vSphere vCenter</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datacenter</td>
<td>Existing folder</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td></td>
<td>Installation program creates the folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Cluster</td>
<td>Always</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Datastore</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere Switch</td>
<td>Always</td>
<td>False</td>
<td><strong>ReadOnly</strong> permission</td>
</tr>
<tr>
<td>vSphere Port Group</td>
<td>Always</td>
<td>False</td>
<td>Listed required privileges</td>
</tr>
<tr>
<td>vSphere vCenter Virtual Machine Folder</td>
<td>Existing folder</td>
<td>True</td>
<td>Listed required privileges</td>
</tr>
</tbody>
</table>

For more information about creating an account with only the required privileges, see *vSphere Permissions and User Management Tasks* in the vSphere documentation.

**Using OpenShift Container Platform with vMotion**

**IMPORTANT**

OpenShift Container Platform generally supports compute-only vMotion. Using Storage vMotion can cause issues and is not supported.

If you are using vSphere volumes in your pods, migrating a VM across datastores either manually or through Storage vMotion causes invalid references within OpenShift Container Platform persistent volume (PV) objects. These references prevent affected pods from starting up and can result in data loss.
Similarly, OpenShift Container Platform does not support selective migration of VMDKs across datastores, using datastore clusters for VM provisioning or for dynamic or static provisioning of PVs, or using a datastore that is part of a datastore cluster for dynamic or static provisioning of PVs.

Cluster resources
When you deploy an OpenShift Container Platform cluster that uses installer-provisioned infrastructure, the installation program must be able to create several resources in your vCenter instance.

A standard OpenShift Container Platform installation creates the following vCenter resources:

- 1 Folder
- 1 Tag category
- 1 Tag
- Virtual machines:
  - 1 template
  - 1 temporary bootstrap node
  - 3 control plane nodes
  - 3 compute machines

Although these resources use 856 GB of storage, the bootstrap node is destroyed during the cluster installation process. A minimum of 800 GB of storage is required to use a standard cluster.

If you deploy more compute machines, the OpenShift Container Platform cluster will use more storage.

Cluster limits
Available resources vary between clusters. The number of possible clusters within a vCenter is limited primarily by available storage space and any limitations on the number of required resources. Be sure to consider both limitations to the vCenter resources that the cluster creates and the resources that you require to deploy a cluster, such as IP addresses and networks.

Networking requirements
You must use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN. You cannot scale the cluster using a second VLAN as a Day 2 operation. The VM in your restricted network must have access to vCenter so that it can provision and manage nodes, persistent volume claims (PVCs), and other resources. Additionally, you must create the following networking resources before you install the OpenShift Container Platform cluster:

**NOTE**

It is recommended that each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server that is discoverable via DHCP. Installation is possible without an NTP server. However, asynchronous server clocks will cause errors, which NTP server prevents.

Required IP Addresses
An installer-provisioned vSphere installation requires two static IP addresses:

- The API address is used to access the cluster API.
The Ingress address is used for cluster ingress traffic.

You must provide these IP addresses to the installation program when you install the OpenShift Container Platform cluster.

DNS records
You must create DNS records for two static IP addresses in the appropriate DNS server for the vCenter instance that hosts your OpenShift Container Platform cluster. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify when you install the cluster. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 12.23. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API VIP</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>This DNS A/AAAA or CNAME record must point to the load balancer for the control plane machines. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Ingress VIP</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>A wildcard DNS A/AAAA or CNAME record that points to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. This record must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>

12.4.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N " \n   -f <path>/<file_name> 1
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**
   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   
   Example output
   ```

   ```
   Agent pid 31874
   ```

   **NOTE**
   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name> 1
   
   Example output
   ```

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.
12.4.8. Adding vCenter root CA certificates to your system trust

Because the installation program requires access to your vCenter’s API, you must add your vCenter’s trusted root CA certificates to your system trust before you install an OpenShift Container Platform cluster.

Procedure

1. From the vCenter home page, download the vCenter’s root CA certificates. Click **Download trusted root CA certificates** in the vSphere Web Services SDK section. The `<vCenter>/certs/download.zip` file downloads.

2. Extract the compressed file that contains the vCenter root CA certificates. The contents of the compressed file resemble the following file structure:

   ```
   certs
   └── lin
       ├── 108f4d17.0
       │    └── 108f4d17.r1
       │         └── 7e757f6a.0
       │             └── 8e4f8471.0
       │                 └── 8e4f8471.r0
       └── mac
           ├── 108f4d17.0
           │    └── 108f4d17.r1
           │         └── 7e757f6a.0
           │             └── 8e4f8471.0
           │                 └── 8e4f8471.r0
           └── win
               ├── 108f4d17.0.crt
               │    └── 108f4d17.r1.crt
               │         └── 7e757f6a.0.crt
               │             └── 8e4f8471.0.crt
               │                 └── 8e4f8471.r0.crt
   3 directories, 15 files
   ```

3. Add the files for your operating system to the system trust. For example, on a Fedora operating system, run the following command:

   ```
   # cp certs/lin/* /etc/pki/ca-trust/source/anchors
   ```

4. Update your system trust. For example, on a Fedora operating system, run the following command:

   ```
   # update-ca-trust extract
   ```

12.4.9. Creating the RHCOS image for restricted network installations

Download the Red Hat Enterprise Linux CoreOS (RHCOS) image to install OpenShift Container Platform on a restricted network VMware vSphere environment.

Prerequisites
Obtain the OpenShift Container Platform installation program. For a restricted network installation, the program is on your mirror registry host.

**Procedure**


2. Under **Version**, select the most recent release of OpenShift Container Platform 4.6 for RHEL 8.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available.

3. Download the **Red Hat Enterprise Linux CoreOS (RHCOS) - vSphere** image.

4. Upload the image you downloaded to a location that is accessible from the bastion server.

   The image is now available for a restricted installation. Note the image name or location for use in OpenShift Container Platform deployment.

### 12.4.10. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on VMware vSphere.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.

- Have the **imageContentSources** values that were generated during mirror registry creation.

- Obtain the contents of the certificate for your mirror registry.

- Retrieve a Red Hat Enterprise Linux CoreOS (RHCOS) image and upload it to an accessible location.

**Procedure**

1. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

   ```bash
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   **Note:** For `<installation_directory>`, specify the directory name to store the files that the installation program creates.
IMPORTANT

Specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

ii. Select vsphere as the platform to target.

iii. Specify the name of your vCenter instance.

iv. Specify the user name and password for the vCenter account that has the required permissions to create the cluster. The installation program connects to your vCenter instance.

v. Select the datacenter in your vCenter instance to connect to.

vi. Select the default vCenter datastore to use.

vii. Select the vCenter cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

viii. Select the network in the vCenter instance that contains the virtual IP addresses and DNS records that you configured.

ix. Enter the virtual IP address that you configured for control plane API access.

x. Enter the virtual IP address that you configured for cluster ingress.

xi. Enter the base domain. This base domain must be the same one that you used in the DNS records that you configured.

xii. Enter a descriptive name for your cluster. The cluster name must be the same one that you used in the DNS records that you configured.

xiii. Paste the pull secret from the Red Hat OpenShift Cluster Manager.

2. In the install-config.yaml file, set the value of platform.vsphere.clusterOSImage to the image location or name. For example:

```yaml
platform:
  vsphere:
```
3. Edit the `install-config.yaml` file to provide the additional information that is required for an installation in a restricted network.
   
   a. Update the `pullSecret` value to contain the authentication information for your registry:

   ```yaml
   pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>"},"email": "you@example.com"}}
   ```

   For `<mirror_host_name>`, specify the registry domain name that you specified in the certificate for your mirror registry, and for `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

   b. Add the `additionalTrustBundle` parameter and value.

   ```yaml
   additionalTrustBundle: |
   -----BEGIN CERTIFICATE-----
   ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
   -----END CERTIFICATE-----
   ```

   The value must be the contents of the certificate file that you used for your mirror registry, which can be an existing, trusted certificate authority or the self-signed certificate that you generated for the mirror registry.

   c. Add the image content resources, which look like this excerpt:

   ```yaml
   imageContentSources:
   - mirrors:
     - <mirror_host_name>:5000/<repo_name>/release
       source: quay.example.com/openshift-release-dev/ocp-release
     - mirrors:
       - <mirror_host_name>:5000/<repo_name>/release
         source: registry.example.com/ocp/release
   ```

   To complete these values, use the `imageContentSources` that you recorded during mirror registry creation.

4. Make any other modifications to the `install-config.yaml` file that you require. You can find more information about the available parameters in the `Installation configuration parameters` section.

5. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the installation process. If you want to reuse the file, you must back it up now.

### 12.4.10.1. Installation configuration parameters

Before you deploy an OpenShift Container Platform cluster, you provide parameter values to describe
your account on the cloud platform that hosts your cluster and optionally customize your cluster’s platform. When you create the `install-config.yaml` installation configuration file, you provide values for the required parameters through the command line. If you customize your cluster, you can modify the `install-config.yaml` file to provide more details about the platform.

**NOTE**

After installation, you cannot modify these parameters in the `install-config.yaml` file.

**IMPORTANT**

The `openshift-install` command does not validate field names for parameters. If an incorrect name is specified, the related file or object is not created, and no error is reported. Ensure that the field names for any parameters that are specified are correct.

### 12.4.10.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

**Table 12.24. Required parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>The API version for the <code>install-config.yaml</code> content. The current version is v1. The installer may also support older API versions.</td>
<td>String</td>
</tr>
<tr>
<td>baseDomain</td>
<td>The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the <code>baseDomain</code> and <code>metadata.name</code> parameter values that uses the <code>&lt;metadata.name&gt;..&lt;baseDomain&gt;</code> format.</td>
<td>A fully-qualified domain or subdomain name, such as <code>example.com</code>.</td>
</tr>
<tr>
<td>metadata</td>
<td>Kubernetes resource <code>ObjectMeta</code>, from which only the <code>name</code> parameter is consumed.</td>
<td>Object</td>
</tr>
<tr>
<td>metadata.name</td>
<td>The name of the cluster. DNS records for the cluster are all subdomains of <code>{{.metadata.name}}.{{.baseDomain}}</code>.</td>
<td>String of lowercase letters, hyphens (−), and periods (.), such as <code>dev</code>.</td>
</tr>
</tbody>
</table>
The configuration for the specific platform upon which to perform the installation: **aws, baremetal, azure, openstack, ovirt, vsphere**. For additional information about **platform.<platform>** parameters, consult the following table for your specific platform.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>platform</strong></td>
<td>The configuration for the specific platform upon which to perform the installation: <strong>aws, baremetal, azure, openstack, ovirt, vsphere</strong>. For additional information about <strong>platform.&lt;platform&gt;</strong> parameters, consult the following table for your specific platform.</td>
<td><strong>Object</strong></td>
</tr>
</tbody>
</table>
| **pullSecret** | Get a pull secret from the Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io. | ```json
{
  "auths":{
    "cloud.openshift.com":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    },
    "quay.io":{
      "auth":"b3Blb=",
      "email":"you@example.com"
    }
  }
}
``` |

### 12.4.10.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

**Table 12.25. Network parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
<td><strong>Values</strong></td>
</tr>
</tbody>
</table>
### networking

The configuration for the cluster network.

**Object**

**NOTE**

You cannot modify parameters specified by the `networking` object after installation.

#### networking.network

**Type**

The cluster network provider Container Network Interface (CNI) plug-in to install.

Either `OpenShiftSDN` or `OVNKubernetes`. The default value is `OpenShiftSDN`.

#### networking.clusterNetwork

The IP address blocks for pods.

The default value is `10.128.0.0/14` with a host prefix of `/23`.

If you specify multiple IP address blocks, the blocks must not overlap.

An array of objects. For example:

```
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
```

#### networking.clusterNetwork.cidr

Required if you use `networking.clusterNetwork`. An IP address block.

An IPv4 network.

#### networking.clusterNetwork.hostPrefix

The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to `23` then each node is assigned a `/23` subnet out of the given `cidr`. A `hostPrefix` value of `23` provides 510 (`2^(32 - 23) - 2`) pod IP addresses.

A subnet prefix.

The default value is `23`.

#### networking.serviceNetwork

The IP address block for services. The default value is `172.30.0.0/16`.

The OpenShift SDN and OVN-Kubernetes network providers support only a single IP address block for the service network.

An array with an IP address block in CIDR format. For example:

```
networking:
  serviceNetwork:
    - 172.30.0.0/16
```
### Networking.MachineNetwork

The IP address blocks for machines. If you specify multiple IP address blocks, the blocks must not overlap.

An array of objects. For example:

```yaml
networking:
  machineNetwork:
    - cidr: 10.0.0.0/16
```

#### Networking.MachineNetwork.cidr

Required if you use `networking.machineNetwork`. An IP address block. The default value is `10.0.0.0/16` for all platforms other than libvirt. For libvirt, the default value is `192.168.126.0/24`.

An IP network block in CIDR notation. For example, `10.0.0.0/16`.

**NOTE**

Set the `networking.machineNetwork` to match the CIDR that the preferred NIC resides in.

---

### 12.4.10.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

**Table 12.26. Optional parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalTrustBundle</td>
<td>A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.</td>
<td>String</td>
</tr>
<tr>
<td>compute</td>
<td>The configuration for the machines that comprise the compute nodes.</td>
<td>Array of machine-pool objects. For details, see the following &quot;Machine-pool&quot; table.</td>
</tr>
<tr>
<td>compute.architecture</td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
Whether to enable or disable simultaneous multithreading, or *hyperthreading*, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>compute.hyperthreading</code></td>
<td>Whether to enable or disable simultaneous multithreading, or <em>hyperthreading</em>, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.</td>
<td></td>
</tr>
<tr>
<td><code>compute.name</code></td>
<td>Required if you use <code>compute</code>. The name of the machine pool.</td>
<td><code>worker</code></td>
</tr>
<tr>
<td><code>compute.platform</code></td>
<td>Required if you use <code>compute</code>. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the <code>controlPlane.platform</code> parameter value.</td>
<td><code>aws, azure, gcp, openstack, ovirt, vsphere, or{}</code></td>
</tr>
<tr>
<td><code>compute.replicas</code></td>
<td>The number of compute machines, which are also known as worker machines, to provision.</td>
<td>A positive integer greater than or equal to 2. The default value is 3.</td>
</tr>
<tr>
<td><code>controlPlane</code></td>
<td>The configuration for the machines that comprise the control plane.</td>
<td>Array of <code>MachinePool</code> objects. For details, see the following “Machine-pool” table.</td>
</tr>
<tr>
<td><code>controlPlane.architecture</code></td>
<td>Determines the instruction set architecture of the machines in the pool. Currently, heterogeneous clusters are not supported, so all pools must specify the same architecture. Valid values are <code>amd64</code> (the default).</td>
<td>String</td>
</tr>
</tbody>
</table>
### controlPlane.hypertreading
Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.

**IMPORTANT**
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>controlPlane.hypertreading</td>
<td>Whether to enable or disable simultaneous multithreading, or hyperthreading, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores.</td>
<td>Enabled or Disabled</td>
</tr>
<tr>
<td>controlPlane.name</td>
<td>Required if you use controlPlane. The name of the machine pool.</td>
<td>master</td>
</tr>
<tr>
<td>controlPlane.platform</td>
<td>Required if you use controlPlane. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the compute.platform parameter value.</td>
<td>aws, azure, gcp, openstack, ovirt, vsphere, or {}</td>
</tr>
<tr>
<td>controlPlane.replicas</td>
<td>The number of control plane machines to provision.</td>
<td>The only supported value is 3, which is the default value.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>credentialsMode</td>
<td>The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported.</td>
<td>Mint, Passthrough, Manual, or an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

**NOTE**

Not all CCO modes are supported for all cloud providers. For more information on CCO modes, see the Cloud Credential Operator entry in the Red Hat Operators reference content.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>fips</strong></td>
<td>Enable or disable FIPS mode. The default is false (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.</td>
<td><strong>false</strong> or <strong>true</strong></td>
</tr>
<tr>
<td><strong>imageContentSources</strong></td>
<td>Sources and repositories for the release-image content.</td>
<td>Array of objects. Includes a <strong>source</strong> and, optionally, <strong>mirrors</strong>, as described in the following rows of this table.</td>
</tr>
<tr>
<td><strong>imageContentSources.source</strong></td>
<td>Required if you use <strong>imageContentSources</strong>. Specify the repository that users refer to, for example, in image pull specifications.</td>
<td>String</td>
</tr>
<tr>
<td><strong>imageContentSources.mirrors</strong></td>
<td>Specify one or more repositories that may also contain the same images.</td>
<td>Array of strings</td>
</tr>
</tbody>
</table>
How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**publish**

How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.

**Internal** or **External**. The default value is **External**.

Setting this field to **Internal** is not supported on non-cloud platforms.

**IMPORTANT**

If the value of the field is set to **Internal**, the cluster will become non-functional. For more information, refer to [BZ#1953035](#).

**sshKey**

The SSH key or keys to authenticate access your cluster machines.

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

One or more keys. For example:

```plaintext
sshKey:
<key1>
<key2>
<key3>
```

### 12.4.10.14. Additional VMware vSphere configuration parameters

Additional VMware vSphere configuration parameters are described in the following table:

**Table 12.27. Additional VMware vSphere cluster parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>platform.vsphere.vCenter</td>
<td>The fully-qualified hostname or IP address of the vCenter server.</td>
<td>String</td>
</tr>
<tr>
<td>platform.vsphere.username</td>
<td>The user name to use to connect to the vCenter instance with. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Table 12.28. Optional VMware vSphere machine pool parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>platform.vsphere.clusterOSImage</code></td>
<td>The location from which the installer downloads the RHCOS image. You must set this parameter to perform an installation in a restricted network.</td>
<td>An HTTP or HTTPS URL, optionally with a SHA-256 checksum. For example, <code>https://mirror.openshift.com/images/rhcos-&lt;version&gt;-vmware.&lt;architecture&gt;.ova</code>.</td>
</tr>
</tbody>
</table>
### Parameter | Description | Values
--- | --- | ---
`platform.vsphere.os Disk.diskSizeGB` | The size of the disk in gigabytes. | Integer

`platform.vsphere.cpus` | The total number of virtual processor cores to assign a virtual machine. | Integer

`platform.vsphere.coresPerSocket` | The number of cores per socket in a virtual machine. The number of virtual sockets on the virtual machine is `platform.vsphere.cpus/platform.vsphere.coresPerSocket`. The default value is 1 | Integer

`platform.vsphere.memoryMB` | The size of a virtual machine's memory in megabytes. | Integer

#### 12.4.10.2. Sample install-config.yaml file for an installer-provisioned VMware vSphere cluster

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 3
    platform:
      vsphere:
        cpus: 2
        coresPerSocket: 2
        memoryMB: 8196
        osDisk:
          diskSizeGB: 120
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
  platform:
    vsphere:
      cpus: 4
      coresPerSocket: 2
      memoryMB: 16384
      osDisk:
        diskSizeGB: 120
metadata:
  name: cluster
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

Optional: Provide additional configuration for the machine pool parameters for the compute and control plane machines.

The cluster name that you specified in your DNS records.
9. The vSphere cluster to install the OpenShift Container Platform cluster in. The installation program uses the root resource pool of the vSphere cluster as the default resource pool.

10. The location of the Red Hat Enterprise Linux CoreOS (RHCOS) image that is accessible from the bastion server.

11. For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

12. Provide the contents of the certificate file that you used for your mirror registry.

13. Provide the `imageContentSources` section from the output of the command to mirror the repository.

12.4.10.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object’s `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object’s `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>
  httpsProxy: https://<username>:<pswd>@<ip>:<port>
  noProxy: example.com
additionalTrustBundle: |
    -----BEGIN CERTIFICATE-----
```

2018
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE

Only the Proxy object named cluster is supported, and no additional proxies can be created.

12.4.11. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the create cluster command of the installation program only once, during initial installation.

Prerequisites

- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```bash
   $ ./openshift-install create cluster --dir <installation_directory> \
   --log-level=info
   ```

   1. For `<installation_directory>`, specify the location of your customized `./install-config.yaml` file.

2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

   **IMPORTANT**
   
   Use the `openshift-install` command from the bastion hosted in the VMC environment.

   **NOTE**
   
   If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

   When the cluster deployment completes, directions for accessing your cluster, including a link to its web console and credentials for the `kubeadmin` user, display in your terminal.

   **Example output**

   ```
   ... INFO Install complete!
   INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
   INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
   INFO Login to the console with user: "kubeadmin", and password: "4vYBz-Ee6gm-ymBZj-Wi5AL"
   INFO Time elapsed: 36m22s
   ```

   **NOTE**
   
   The cluster access and credential information also outputs to `<installation_directory>/openshift_install.log` when an installation succeeds.
**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**IMPORTANT**

You must not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

### 12.4.12. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (**oc**) in order to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of **oc**.

#### 12.4.12.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```
   $ tar xvzf <file>
   ```

5. Place the **oc** binary in a directory that is on your PATH.

   To check your PATH, execute the following command:

   ```
   $ echo $PATH
   ```
After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```

12.4.12.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (`oc`) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the `oc` binary to a directory that is on your **PATH**.
   
   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
C:\> oc <command>
```

12.4.12.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (`oc`) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the `oc` binary to a directory on your **PATH**.
   
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```
$ oc <command>
```
12.4.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

**Procedure**

1. Export the **kubeadmin** credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   **Example output**

   system:admin

12.4.14. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the **OperatorHub** object:

  ```bash
  $ oc patch OperatorHub cluster --type json \
   -p '["op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
  ```

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the **Administration → Cluster Settings → Global Configuration → OperatorHub** page, click the **Sources** tab, where you can create, delete, disable, and enable individual sources.

12.4.15. Creating registry storage
After you install the cluster, you must create storage for the Registry Operator.

12.4.15.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

12.4.15.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the Recreate rollout strategy during upgrades.

12.4.15.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

Prerequisites

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

IMPORTANT

OpenShift Container Platform supports ReadWriteOnce access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, ReadWriteMany access is required.

- Must have "100Gi" capacity.
IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**
   
   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```
   $ oc get clusteroperator image-registry
   ```

**12.4.16. Telemetry access for OpenShift Container Platform**

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

12.4.17. Next steps

- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the must-gather tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.5. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure that you provision by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

12.5.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.
Configure the following firewall rules:

- An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the Internet. This is used by nodes and applications to download container images.
- An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
- An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

You must have the following information to deploy OpenShift Container Platform:

- The OpenShift Container Platform cluster name, such as vmc-prod-1.
- The base DNS name, such as companyname.com.
- If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to 10.128.0.0/14 and 172.30.0.0/16, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
- The following vCenter information:
  - vCenter hostname, username, and password
  - Datacenter name, such as SDDC-Datacenter
  - Cluster name, such as Cluster-1
  - Network name
  - Datastore name, such as WorkloadDatastore

**NOTE**

It is recommended to move your vSphere cluster to the VMC Compute-ResourcePool resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The openshift-install installation program
    - The OpenShift CLI (oc) tool
NOTE

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.5.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
  - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

12.5.2. vSphere prerequisites

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

12.5.3. Internet access for OpenShift Container Platform
In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 12.5.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

**Table 12.29. Minimum supported vSphere version for VMware components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See *Edit Time Configuration for a Host* in the VMware documentation.
12.5.5. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

12.5.5.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

12.5.5.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

12.5.5.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>
### Machine Specifications

<table>
<thead>
<tr>
<th>Machine</th>
<th>Operating System</th>
<th>vCPU</th>
<th>Virtual RAM</th>
<th>Storage</th>
<th>IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

### 12.5.5.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

### 12.5.6. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

#### Prerequisites

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

#### Procedure

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

### 12.5.6.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in `initramfs` during boot to fetch Ignition config from the machine config server.
During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 12.32. Control plane machines to control plane machines
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**

   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 12.33. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Internal</th>
<th>External</th>
<th>Back-end machines (pool members)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
NOTE

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer** Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>x</td>
<td>x</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>x</td>
<td>x</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for *Configuring chrony time service*.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**12.5.6.2. User-provisioned DNS requirements**
DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 12.35. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes</td>
<td><code>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td><code>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Routes</td>
<td><code>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td><code>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Master hosts</td>
<td><code>&lt;master&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</code></td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>
Worker hosts

Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

### Example 12.9. Sample DNS zone database

```bash
$TTL 1W
@ IN SOA ns1.example.com. root (2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
api.octn IN A 192.168.1.5
api-int.octn IN A 192.168.1.5
;
*.apps.octn IN A 192.168.1.5
;
bootstrap.octn IN A 192.168.1.96
;
master0.octn IN A 192.168.1.97
master1.octn IN A 192.168.1.98
master2.octn IN A 192.168.1.99
;
worker0.octn IN A 192.168.1.11
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 12.10. Sample DNS zone database for reverse records

```
$TTL 1W
@ IN SOA ns1.example.com. root (  
2019070700 ; serial  
3H ; refresh (3 hours)  
30M ; retry (30 minutes)  
2W ; expiry (2 weeks)  
1W ) ; minimum (1 week)
IN NS ns1.example.com.  
; The syntax is "last octet" and the host must have an FQDN  
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;  
96 IN PTR bootstrap.ocp4.example.com.
;  
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;  
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;  
;EOF
```

12.5.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N "" \n   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   **Example output**

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

**Next steps**

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster's machines.
12.5.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   IMPORTANT

   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   IMPORTANT

   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

12.5.9. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.
Procedure

1. Create an installation directory to store your required installation assets in:

   ```bash
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**
   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**
   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**
   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

### 12.5.9.1. Sample `install-config.yaml` file for VMware vSphere

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: test
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
datacenter: datacenter
```
The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

The controlPlane section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the compute section must begin with a hyphen, -, and the first line of the controlPlane section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

Whether to enable or disable simultaneous multithreading, or hyperthreading. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to Disabled. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

You must set the value of the replicas parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.

The fully-qualified hostname or IP address of the vCenter server.

The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

The password associated with the vSphere user.

The vSphere datacenter.

The default vSphere datastore to use.

Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, /<datacenter_name>/vm/<folder_name>/<subfolder_name>. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.
Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the **x86_64** architecture.

The pull secret that you obtained from OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the **core** user in Red Hat Enterprise Linux CoreOS (RHCOS).

### 12.5.9.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

**Prerequisites**

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s **spec.noProxy** field to bypass the proxy if necessary.

**NOTE**

The Proxy object **status.noProxy** field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object **status.noProxy** field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port>  # 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port>  # 2
  noProxy: example.com  # 3
additionalTrustBundle: |  # 4
```
A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.

A proxy URL to use for creating HTTPS connections outside the cluster.

A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use `*` to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil `spec`.

NOTE

Only the Proxy object named `cluster` is supported, and no additional proxies can be created.

12.5.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.

- You created the install-config.yaml installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory> ①
   ```

   Example output

   ```bash
   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift
   ```

   ① For `<installation_directory>`, specify the installation directory that contains the install-config.yaml file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```bash
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.

   - You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the mastersSchedulable parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to false. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory>  
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```
   ├── auth
   │    └── kubeadmin-password
   │    └── kubeconfig
   │    └── bootstrap.ign
   │    └── master.ign
   │    └── metadata.json
   │    └── worker.ign
   ```

12.5.11. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

**Prerequisites**

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

**Procedure**

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  ```bash
  $ jq -r .infraID <installation_directory>/metadata.json  
  ```

  For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

  **Example output**

  ```
  openshift-vw9j6  
  ```
The output of this command is your cluster name and a random string.

12.5.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a vSphere cluster.

Procedure

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

   ```json
   
   "ignition": {
     "config": {
       "merge": [
         {
           "source": "<bootstrap_ignition_config_url>"
         },
         "verification": {}  
       ]
     },
     "timeout": {},
     "version": "3.1.0"
   },
   "networkd": {},
   "passwd": {},
   "storage": {},
   "systemd": {}
   }
   ``

   Specify the URL of the bootstrap Ignition config file that you hosted.

   When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

   - `<installation_directory>/master.ign`
   - `<installation_directory>/worker.ign`
4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   a. Click the VMs and Templates view.
   b. Right-click the name of your datacenter.
   c. Click New Folder → New VM and Template Folder.
   d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the `install-config.yaml` file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.

7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.
a. From the **Hosts and Clusters** tab, right-click your cluster name and select **Deploy OVF Template**.

b. On the **Select an OVF** tab, specify the name of the RHCOS OVA file that you downloaded.

c. On the **Select a name and folder** tab, set a **Virtual machine name** for your template, such as **Template-RHCOS**. Click the name of your vSphere cluster and select the folder you created in the previous step.

d. On the **Select a compute resource** tab, click the name of your vSphere cluster.

e. On the **Select storage** tab, configure the storage options for your VM.
   - Select **Thin Provision** or **Thick Provision**, based on your storage preferences.
   - Select the datastore that you specified in your **install-config.yaml** file.

f. On the **Select network** tab, specify the network that you configured for the cluster, if available.

g. When creating the OVF template, do not specify values on the **Customize template** tab or configure the template any further.

**IMPORTANT**

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.

a. Right-click the template name and click **Clone → Clone to Virtual Machine**

b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **control-plane-0** or **compute-1**.

c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.

d. On the **Select a compute resource** tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

e. Optional: On the **Select storage** tab, customize the storage options.

f. On the **Select clone options**, select **Customize this virtual machine’s hardware**

g. On the **Customize hardware** tab, click **VM Options → Advanced**.
   - Optional: Override default DHCP networking in vSphere. To enable static IP networking:
     i. Set your static IP configuration:

```
$ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>:none
nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]
```
Example command

```bash
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"

$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.net-"work-kargs=${IPCFG}"
```

ii. Set the **guestinfo.afterburn.initrd.network-kargs** property before booting a VM from an OVA in vSphere:

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM's CPU and memory reservation have the following values:
  - Memory reservation value must be equal to its configured memory size.
  - CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.
- Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
  - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
  - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
  - **disk.EnableUUID**: Specify **TRUE**.
- In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.
- Complete the configuration and power on the VM.

9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

12.5.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

**Prerequisites**

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

**Procedure**

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click **Clone → Clone to Virtual Machine**
   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.
   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
   d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
   e. Optional: On the **Select storage** tab, customize the storage options.
   f. On the **Select clone options**, select **Customize this virtual machine’s hardware**
   g. On the **Customize hardware** tab, click **VM Options → Advanced**.
      - From the **Latency Sensitivity** list, select **High**.
      - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
        - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
        - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
        - **disk.EnableUUID**: Specify **TRUE**.
   h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under **Add network adapter** if there are multiple networks available.
   i. Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

**12.5.14. Disk partitioning**

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making **/var** or a subdirectory of **/var**, such as **/var/lib/etcd**, a separate partition, but not both.
IMPORTANT

Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

Creating a separate `/var` partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadmin-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...  
   ```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage
device on the **worker** systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```
apiVersion: machineconfiguration.openshift.io/v1
description: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  
          partitions:
            - label: var
              startMiB: <partition_start_offset>  
              sizeMiB: <partition_size>  
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
  systemd:
    units:
      - name: var.mount
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota
          [Install]
          WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstallations of RHCOS might overwrite the beginning of the data partition.
3. The size of the data partition in mebibytes.
4. The name of the mount unit must match the directory specified in the **Where** directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-lib-containers.mount`.
5. The **prjquota** mount option must be enabled for filesystems used for container storage.
NOTE
When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

   ```bash
   $ openshift-install create ignition-configs --dir $HOME/clusterconfig
   $ ls $HOME/clusterconfig/
   auth bootstrap.ign master.ign metadata.json worker.ign
   ```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

12.5.15. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (`oc`) in order to interact with OpenShift Container Platform from a command-line interface. You can install `oc` on Linux, Windows, or macOS.

**IMPORTANT**

If you installed an earlier version of `oc`, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of `oc`.

12.5.15.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (`oc`) binary on Linux by using the following procedure.

**Procedure**


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   ```bash
   $ tar xvf <file>
   ```

5. Place the `oc` binary in a directory that is on your PATH.
   To check your PATH, execute the following command:

   ```bash
   $ echo $PATH
   ```

After you install the OpenShift CLI, it is available using the `oc` command:

```bash
$ oc <command>
```
12.5.15.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the **oc** binary to a directory that is on your **PATH**.

   To check your **PATH**, open the command prompt and execute the following command:

   ```shell
   C:\> path
   ```

   After you install the OpenShift CLI, it is available using the **oc** command:

   ```shell
   C:\> oc <command>
   ```

12.5.15.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

**Procedure**


2. Select the appropriate version in the **Version** drop-down menu.

3. Click **Download Now** next to the **OpenShift v4.6 MacOSX Client** entry and save the file.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your **PATH**.

   To check your **PATH**, open a terminal and execute the following command:

   ```shell
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the **oc** command:

   ```shell
   $ oc <command>
   ```

12.5.16. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.
Prerequisites

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \  
   --log-level=info

   1 For `<installation_directory>`, specify the path to the directory that you stored the  
   installation files in.
   2 To view different installation details, specify warn, debug, or error instead of info.

Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped  
on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   IMPORTANT
   
   You must remove the bootstrap machine from the load balancer at this point.  
   You can also remove or reformat the machine itself.

12.5.17. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster kubeconfig file. The  
kubeconfig file contains information about the cluster that is used by the CLI to connect a client to the  
correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container  
Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the oc CLI.

Procedure
1. Export the `kubeadmin` credentials:

```bash
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

```bash
$ oc whoami
```

Example output

```
system:admin
```

### 12.5.18. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

```bash
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

The output lists all of the machines that you created.

**NOTE**

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```bash
$ oc get csr
```

Example output
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in Pending status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the machine-approver if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the oc exec, oc rsh, and oc logs commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the node-bootstrapper service account in the system:node or system:admin groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  $$ $ oc adm certificate approve <csr_name> $$

  - **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  $$ $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve $$

**NOTE**

Some Operators might not become available until some CSRs are approved.
4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name>  
```

<csr_name> is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see [Certificate Signing Requests](#).
12.5.19. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

```bash
$ watch -n5 oc get clusteroperators
```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>openshift-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

12.5.19.1. Image registry removed during installation
On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as **Removed**. This allows **openshift-installer** to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the **managementState** from **Removed** to **Managed**.

**NOTE**

The Prometheus console provides an **ImageRegistryRemoved** alert, for example:

"Image Registry has been removed. **ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected. Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

### 12.5.19.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

### 12.5.19.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.
IMPORTANT

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

Procedure

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   NOTE

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```

   NOTE

   If the storage type is `emptyDir`, the replica number cannot be greater than 1.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   Example output

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```
   $ oc get clusteroperator image-registry
   ```

12.5.19.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.
Procedure

- To set the image registry storage to an empty directory:

```bash
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"storage":{"emptyDir":{}}}'}
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

### 12.5.19.2.3. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the *Recreate* rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the *Recreate* rollout strategy and runs with only 1 replica:

```bash
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere `PersistentVolumeClaim` object:

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage
  namespace: openshift-image-registry
spec:
```

2062
A unique name that represents the `PersistentVolumeClaim` object.

The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.

The access mode of the persistent volume claim. With `ReadWriteOnce`, the volume can be mounted with read and write permissions by a single node.

The size of the persistent volume claim.

b. Create the `PersistentVolumeClaim` object from the file:

```bash
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. Edit the registry configuration so that it references the correct PVC:

```bash
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

**Example output**

```yaml
storage:
  pvc:
    claim: 1
```

Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring the registry for vSphere](#).

### 12.5.20. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

```bash
$ watch -n5 oc get clusteroperators
```
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```
   $ oc get pods --all-namespaces
   ```

   Example output

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 1 9m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-67b9g</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 3m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-ljcmx</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 1m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>apiserver-z25h4</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 2m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-authentication-operator</td>
<td>authentication-operator-69d5d8bf84-vh2n8</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td></td>
<td>Running 0 5m</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```
   $ oc logs <pod_name> -n <namespace>
   ```

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.
12.5.21. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

12.5.22. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

12.5.23. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.6. INSTALLING A CLUSTER ON VMC WITH USER-PROVISIONED INFRASTRUCTURE AND NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.6, you can install a cluster on your VMware vSphere instance using infrastructure you provision with customized network configuration options by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.
By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing VXLAN configurations. You must set most of the network configuration parameters during installation, and you can modify only `kubeProxy` configuration parameters in a running cluster.

### 12.6.1. Setting up VMC for vSphere

You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

![OpenShift](image)

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Configure the following firewall rules:
  - An ANY:ANY firewall rule between the OpenShift Container Platform compute network and the Internet. This is used by nodes and applications to download container images.
  - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
  - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

- You must have the following information to deploy OpenShift Container Platform:
  - The OpenShift Container Platform cluster name, such as `vmc-prod-1`.
  - The base DNS name, such as `companyname.com`.
  - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to `10.128.0.0/14` and `172.30.0.0/16`, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
  - The following vCenter information:
- vCenter hostname, username, and password
- Datacenter name, such as SDDC-Datacenter
- Cluster name, such as Cluster-1
- Network name
- Datastore name, such as WorkloadDatastore

**NOTE**

It is recommended to move your vSphere cluster to the VMC Compute-ResourcePool resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The openshift-install installation program
    - The OpenShift CLI (oc) tool

**NOTE**

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

### 12.6.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
Storage requirements

- vCPUs
- vRAM
- Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.

12.6.2. vSphere prerequisites

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.
- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to access Red Hat Insights.

12.6.3. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

12.6.4. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

Table 12.36. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>Minimum supported versions</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.

**IMPORTANT**

You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

### 12.6.5. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

#### 12.6.5.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.
The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

12.6.5.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

12.6.5.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: \((\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}\).

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

12.6.5.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

12.6.6. Creating the user-provisioned infrastructure
Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

Prerequisites

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.

Procedure

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

12.6.6.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 12.37. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td>Protocol</td>
<td>Port</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

**Table 12.38. All machines to control plane**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

**Table 12.39. Control plane machines to control plane machines**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

**Network topology requirements**
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

**Load balancers**
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer:** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.
IMPORTANT
Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

Table 12.40. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

NOTE
The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. Application Ingress load balancer: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

Table 12.41. Application Ingress load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
</tbody>
</table>
The machines that run the Ingress router pods, compute, or worker, by default.

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**12.6.6.2. User-provisioned DNS requirements**

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>.

**Table 12.42. Required DNS records**

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;..&lt;base_domain&gt;</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
</tbody>
</table>
Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;</strong></td>
<td></td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td><strong>Routes</strong></td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td><strong>Bootstrap</strong></td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td><strong>Master hosts</strong></td>
<td>&lt;master&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td><strong>Worker hosts</strong></td>
<td>&lt;worker&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 12.11. Sample DNS zone database**

```
$TTL 1W
@ IN SOA ns1.example.com. root (  
 2019070700 ; serial 
 3H ; refresh (3 hours)
```

---

OpenShift Container Platform 4.6 Installing
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 12.12. Sample DNS zone database for reverse records

```plaintext
$TTL 1W
@ IN SOA ns1.example.com. root ( 2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week) IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
```

The syntax is "last octet" and the host must have an FQDN with a trailing dot.
12.6.7. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your `ssh-agent` and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user `core`. When you deploy the cluster, the key is added to the `core` user’s `~/.ssh/authorized_keys` list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" \\
-f <path>/<file_name>  
```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the `ssh-agent` process as a background task:
NOTE
If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

```
$ ssh-add <path>/<file_name>
```

Example output

```
Identity added: /home/<you>/<path>/<file_name> (computer_name)
```

Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

12.6.8. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on a local computer.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space

Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

IMPORTANT

The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.
Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar xvf openshift-install-linux.tar.gz
   ```

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

### 12.6.9. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.
IMPORTANT

The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

12.6.9.1. Sample `install-config.yaml` file for VMware vSphere

You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: test
platform:
  vsphere:
    vcenter: your.vcenter.server
    username: username
    password: password
datacenter: datacenter
defaultDatastore: datastore
  folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"
fips: false
pullSecret: '{"auths":...}'
sshKey: 'ssh-ed25519 AAAA...
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2. The `controlPlane` section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3. Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.
If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

You must set the value of the `replicas` parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

The number of control plane machines that you add to the cluster. Because the cluster uses these values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.

The fully-qualified hostname or IP address of the vCenter server.

The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

The password associated with the vSphere user.

The vSphere datacenter.

The default vSphere datastore to use.

Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret that you obtained from OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

### 12.6.9.2. Configuring the cluster-wide proxy during installation

2082
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port> ¹
     httpsProxy: https://<username>:<pswd>@<ip>:<port> ²
     noProxy: example.com ³
     additionalTrustBundle: |
       -----BEGIN CERTIFICATE-----
       <MY_TRUSTED_CA_CERT>
       -----END CERTIFICATE-----
       ...
   
   ¹ A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.
   ² A proxy URL to use for creating HTTPS connections outside the cluster.
   ³ A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.
   ⁴ If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the `Proxy` object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network
Operator then creates a `trusted-ca-bundle` config map that merges the contents specified for the `trustedCA` parameter with the RHCOS trust bundle. The `additionalTrustBundle` field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy `readinessEndpoints` field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named `cluster` that uses the proxy settings in the provided `install-config.yaml` file. If no proxy settings are provided, a `cluster Proxy` object is still created, but it will have a nil `spec`.

**NOTE**

Only the `Proxy` object named `cluster` is supported, and no additional proxies can be created.

### 12.6.10. Specifying advanced network configuration

You can use advanced configuration customization to integrate your cluster into your existing network environment by specifying additional configuration for your cluster network provider. You can specify advanced network configuration only before you install the cluster.

**IMPORTANT**

Modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

**Prerequisites**

- Create the `install-config.yaml` file and complete any modifications to it.
- Create the Ignition config files for your cluster.

**Procedure**

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   `<installation_directory>`

   Specifies the name of the directory that contains the `install-config.yaml` file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named `cluster-network-03-config.yml` in the `<installation_directory>/manifests/` directory:

   ```
where:

<installation_directory>

Specifies the directory name that contains the manifests/ directory for your cluster.

3. Open the cluster-network-03-config.yml file in an editor and specify the advanced network configuration for your cluster, such as in the following example:

**Specify a different VXLAN port for the OpenShift SDN network provider**

```yaml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
defaultNetwork:
  openshiftSDNConfig:
    vxlanPort: 4800
```

4. Save the cluster-network-03-config.yml file and quit the text editor.

5. Optional: Back up the manifests/cluster-network-03-config.yml file. The installation program deletes the manifests/ directory when creating the cluster.

6. Remove the Kubernetes manifest files that define the control plane machines and compute machineSets:

```bash
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the MachineSet files to create compute machines by using the machine API, but you must update references to them to match your environment.

**12.6.11. Cluster Network Operator configuration**

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named `cluster`. The CR specifies the fields for the Network API in the operator.openshift.io API group.

The CNO configuration inherits the following fields during cluster installation from the Network API in the Network.config.openshift.io API group and these fields cannot be changed:

**clusterNetwork**

- IP address pools from which pod IP addresses are allocated.
You can specify the cluster network provider configuration for your cluster by setting the fields for the `defaultNetwork` object in the CNO object named `cluster`.

### 12.6.11.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

**Table 12.43. Cluster Network Operator configuration object**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata.name</td>
<td>string</td>
<td>The name of the CNO object. This name is always <code>cluster</code>.</td>
</tr>
<tr>
<td>spec.clusterNetwork</td>
<td>array</td>
<td>A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>clusterNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.0.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- cidr: 10.128.32.0/19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hostPrefix: 23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the <code>install-config.yaml</code> file.</td>
</tr>
<tr>
<td>spec.serviceNetwork</td>
<td>array</td>
<td>A block of IP addresses for services. The OpenShift SDN and OVN-Kubernetes Container Network Interface (CNI) network providers support only a single IP address block for the service network. For example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>spec:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>serviceNetwork:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- 172.30.0.0/14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is ready-only and specified in the <code>install-config.yaml</code> file.</td>
</tr>
<tr>
<td>spec.defaultNetwork</td>
<td>object</td>
<td>Configures the Container Network Interface (CNI) cluster network provider for the cluster network.</td>
</tr>
<tr>
<td>spec.kubeProxyConfig</td>
<td>object</td>
<td>The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network provider, the kube-proxy configuration has no effect.</td>
</tr>
</tbody>
</table>
defaultNetwork object configuration
The values for the defaultNetwork object are defined in the following table:

Table 12.44. defaultNetwork object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>Either OpenShiftSDN or OVNKubernetes. The cluster network provider is selected during installation. This value cannot be changed after cluster installation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>OpenShift Container Platform uses the OpenShift SDN Container Network Interface (CNI) cluster network provider by default.</td>
</tr>
<tr>
<td>openshiftSDNConfig</td>
<td>object</td>
<td>This object is only valid for the OpenShift SDN cluster network provider.</td>
</tr>
<tr>
<td>ovnKubernetesConfig</td>
<td>object</td>
<td>This object is only valid for the OVN-Kubernetes cluster network provider.</td>
</tr>
</tbody>
</table>

Configuration for the OpenShift SDN CNI cluster network provider
The following table describes the configuration fields for the OpenShift SDN Container Network Interface (CNI) cluster network provider.

Table 12.45. openshiftSDNConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>string</td>
<td>Configures the network isolation mode for OpenShift SDN. The default value is NetworkPolicy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The values Multitenant and Subnet are available for backwards compatibility with OpenShift Container Platform 3.x but are not recommended. This value cannot be changed after cluster installation.</td>
</tr>
</tbody>
</table>
The maximum transmission unit (MTU) for the VXLAN overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to **50** less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of **9001**, and some have an MTU of **1500**, you must set this value to **1450**.

This value cannot be changed after cluster installation.

The port to use for all VXLAN packets. The default value is **4789**. This value cannot be changed after cluster installation.

If you are running in a virtualized environment with existing nodes that are part of another VXLAN network, then you might be required to change this. For example, when running an OpenShift SDN overlay on top of VMware NSX-T, you must select an alternate port for the VXLAN, because both SDNs use the same default VXLAN port number.

On Amazon Web Services (AWS), you can select an alternate port for the VXLAN between port **9000** and port **9999**.

**Example OpenShift SDN configuration**

```yaml
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

**Configuration for the OVN-Kubernetes CNI cluster network provider**

The following table describes the configuration fields for the OVN-Kubernetes CNI cluster network provider.

**Table 12.46. ovnKubernetesConfig object**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU.

If the auto-detected value is not what you expected it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes.

If your cluster requires different MTU values for different nodes, you must set this value to 100 less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of 9001, and some have an MTU of 1500, you must set this value to 1400.

This value cannot be changed after cluster installation.

The port to use for all Geneve packets. The default value is 6081. This value cannot be changed after cluster installation.

---

### Example OVN-Kubernetes configuration

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
```

### kubeProxyConfig object configuration

The values for the `kubeProxyConfig` object are defined in the following table:

#### Table 12.47. kubeProxyConfig object

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iptablesSyncPeriod</td>
<td>string</td>
<td>The refresh period for <em>iptables</em> rules. The default value is 30s. Valid suffixes include <em>s</em>, <em>m</em>, and <em>h</em> and are described in the <em>Go time</em> package documentation.</td>
</tr>
<tr>
<td>proxyArguments.iptables-min-sync-period</td>
<td>array</td>
<td>The minimum duration before refreshing <em>iptables</em> rules. This field ensures that the refresh does not happen too frequently. Valid suffixes include <em>s</em>, <em>m</em>, and <em>h</em> and are described in the <em>Go time</em> package. The default value is: kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s</td>
</tr>
</tbody>
</table>

**12.6.12. Creating the Ignition config files**

Because you must manually start the cluster machines, you must generate the Ignition config files that the cluster needs to make its machines.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending *node-bootstrapper* certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
Procedure

- Obtain the Ignition config files:

  $ ./openshift-install create ignition-configs --dir <installation_directory> ①

  For `<installation_directory>`, specify the directory name to store the files that the installation program creates.

  **IMPORTANT**
  
  If you created an `install-config.yaml` file, specify the directory that contains it. Otherwise, specify an empty directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

  The following files are generated in the directory:

  - `auth`
    - `kubeadmin-password`
    - `kubeconfig`
    - `bootstrap.ign`
    - `master.ign`
    - `metadata.json`
    - `worker.ign`

12.6.13. Extracting the infrastructure name

The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

Prerequisites

- You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.
- You generated the Ignition config files for your cluster.
- You installed the `jq` package.

Procedure

- To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

  $ jq -r .infraID <installation_directory>/metadata.json ①
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

Example output

```
openshift-vw9j6
```

The output of this command is your cluster name and a random string.

12.6.14. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a vSphere cluster.

Procedure

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>",
          "verification": {}
        }
      ],
      "timeouts": {},
      "version": "3.1.0"
    },
    "networkd": {},
    "passwd": {},
    "storage": {},
    "systemd": {}
  }
}
```

Specify the URL of the bootstrap Ignition config file that you hosted.
When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

   - `<installation_directory>/master.ign`
   - `<installation_directory>/worker.ign`
   - `<installation_directory>/merge-bootstrap.ign`

4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

   ```bash
   $ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
   $ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
   $ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
   ```

   **IMPORTANT**

   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

   The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   
   a. Click the **VMs and Templates** view.
   
   b. Right-click the name of your datacenter.
   
   c. Click **New Folder → New VM and Template Folder**.
   
   d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the `install-config.yaml` file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.
7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.

a. From the Hosts and Clusters tab, right-click your cluster name and select Deploy OVF Template.

b. On the Select an OVF tab, specify the name of the RHCOS OVA file that you downloaded.

c. On the Select a name and folder tab, set a Virtual machine name for your template, such as Template-RHCOS. Click the name of your vSphere cluster and select the folder you created in the previous step.

d. On the Select a compute resource tab, click the name of your vSphere cluster.

e. On the Select storage tab, configure the storage options for your VM.
   - Select Thin Provision or Thick Provision, based on your storage preferences.
   - Select the datastore that you specified in your install-config.yaml file.

f. On the Select network tab, specify the network that you configured for the cluster, if available.

g. When creating the OVF template, do not specify values on the Customize template tab or configure the template any further.

**IMPORTANT**

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.

a. Right-click the template name and click Clone → Clone to Virtual Machine.

b. On the Select a name and folder tab, specify a name for the VM. You might include the machine type in the name, such as control-plane-0 or compute-1.

c. On the Select a name and folder tab, select the name of the folder that you created for the cluster.

d. On the Select a compute resource tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

e. Optional: On the Select storage tab, customize the storage options.

f. On the Select clone options, select Customize this virtual machine’s hardware.
g. On the **Customize hardware** tab, click **VM Options → Advanced**.

- Optional: Override default DHCP networking in vSphere. To enable static IP networking:
  
  i. Set your static IP configuration:

  ```
  $ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>::none
  nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
  
  Example command
  ```

  ```
  $ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0::none
  nameserver=8.8.8.8"
  
  ii. Set the **guestinfo.afterburn.initrd.network-kargs** property before booting a VM from an OVA in vSphere:

  ```
  $ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
  ```

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:

  - Memory reservation value must be equal to its configured memory size.
  - CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.

- Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:

  - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.

  - **guestinfo.ignition.config.data.encoding**: Specify base64.

  - **disk.EnableUUID**: Specify TRUE.

h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.

  i. Complete the configuration and power on the VM.

9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.
12.6.15. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

Prerequisites

- Obtain the base64-encoded Ignition file for your compute machines.
- You have access to the vSphere template that you created for your cluster.

Procedure

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click **Clone → Clone to Virtual Machine**
   b. On the **Select a name and folder** tab, specify a name for the VM. You might include the machine type in the name, such as **compute-1**.
   c. On the **Select a name and folder** tab, select the name of the folder that you created for the cluster.
   d. On the **Select a compute resource** tab, select the name of a host in your datacenter.
   e. Optional: On the **Select storage** tab, customize the storage options.
   f. On the **Select clone options**, select **Customize this virtual machine’s hardware**
   g. On the **Customize hardware** tab, click **VM Options → Advanced**.
      - From the **Latency Sensitivity** list, select **High**.
      - Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
         - **guestinfo.ignition.config.data**: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
         - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
         - **disk.EnableUUID**: Specify **TRUE**.
   h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under **Add network adapter** if there are multiple networks available.
   i. Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

12.6.16. Disk partitioning
In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making /var or a subdirectory of /var, such as /var/lib/etcd, a separate partition, but not both.

**IMPORTANT**

Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to coreos-installer that allow you to retain existing data partitions.

Creating a separate /var partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the /var partition or a subdirectory of /var. For example:

- /var/lib/containers: Holds container-related content that can grow as more images and containers are added to a system.
- /var/lib/etcd: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- /var: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a /var directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because /var must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate /var partition by creating a machine config that is inserted during the openshift-install preparation phases of an OpenShift Container Platform installation.

**Procedure**

1. Create a directory to hold the OpenShift Container Platform installation files:

   $$ mkdir $HOME/clusterconfig$$

2. Run openshift-install to create a set of files in the manifest and openshift subdirectories. Answer the system questions as you are prompted:
3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage device on the `worker` systems, and set the storage size as appropriate. This example places the `/var` directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
custom: MachineConfig
description: 
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  
        partitions:
          - label: var
            startMiB: <partition_start_offset>  
            sizeMiB: <partition_size>  
        filesystems:
          - device: /dev/disk/by-partlabel/var
            path: /var
            format: xfs
  systemd:
    units:
      - name: var.mount
        enabled: true
        contents: |
        [Unit]
        Before=local-fs.target
        [Mount]
        What=/dev/disk/by-partlabel/var
        Where=/var
        Options=defaults,prjquota
        [Install]
        WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.

2. When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future
reinstalls of RHCOS might overwrite the beginning of the data partition.

3. The size of the data partition in mebibytes.

4. The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-lib-containers.mount`.

5. The `prjquota` mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

   ```bash
   $ openshift-install create ignition-configs --dir $HOME/clusterconfig
   $ ls $HOME/clusterconfig/
   auth bootstrap.ign master.ign metadata.json worker.ign
   ``

   Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

12.6.17. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

**Prerequisites**

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

**Procedure**

1. Monitor the bootstrap process:

   ```bash
   $ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \  
   --log-level=info
   ``

   **1** For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **2** To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.  

2099
Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

**IMPORTANT**

You must remove the bootstrap machine from the load balancer at this point. You can also remove or reformat the machine itself.

### 12.6.18. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   **Example output**

   ```
   system:admin
   ```

### 12.6.19. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.
Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   
   ```
   $ oc get nodes
   ```

   **Example output**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  63m  v1.19.0
   master-1  Ready     master  63m  v1.19.0
   master-2  Ready     master  64m  v1.19.0
   ```

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```
   $ oc get csr
   ```

   **Example output**

   ```
   NAME        AGE     REQUESTOR                               CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   ...
   ```

   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:
NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name> 1
  
  1 <csr_name> is the name of a CSR from the list of current CSRs.
  
  - To approve all pending CSRs, run the following command:
  ```

  ```
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

  

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

  ```
  $ oc get csr
  ```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57lv</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 12. INSTALLING ON VMC

5. If the remaining CSRs are not approved, and are in the Pending status, approve the CSRs for
your cluster machines:
To approve them individually, run the following command for each valid CSR:
$ oc adm certificate approve <csr_name> 1
1

<csr_name> is the name of a CSR from the list of current CSRs.

To approve all pending CSRs, run the following command:
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
6. After all client and server CSRs have been approved, the machines have the Ready status.
Verify this by running the following command:
$ oc get nodes

Example output
NAME
master-0
master-1
master-2
worker-0
worker-1

STATUS ROLES AGE VERSION
Ready master 73m v1.20.0
Ready master 73m v1.20.0
Ready master 74m v1.20.0
Ready worker 11m v1.20.0
Ready worker 11m v1.20.0

NOTE
It can take a few minutes after approval of the server CSRs for the machines to
transition to the Ready status.
Additional information
For more information on CSRs, see Certificate Signing Requests .

12.6.20. Initial Operator configuration
After the control plane initializes, you must immediately configure some Operators so that they all
become available.
Prerequisites
Your control plane has initialized.
Procedure
1. Watch the cluster components come online:
$ watch -n5 oc get clusteroperators

2103


Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

12.6.20.1. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator bootstraps itself as Removed. This allows openshift-installer to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the managementState from Removed to Managed.

NOTE

The Prometheus console provides an ImageRegistryRemoved alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing
configs.imageregistry.operator.openshift.io."

2104
12.6.20.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the **Recreate** rollout strategy during upgrades.

12.6.20.2.1. Configuring block registry storage for VMware vSphere

To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only 1 replica:

   ```bash
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
   {"rolloutStrategy":"Recreate","replicas":1}}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: image-registry-storage
   namespace: openshift-image-registry
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
       requests:
         storage: 100Gi
   ```

   **A unique name that represents the `PersistentVolumeClaim` object.**

   **The namespace for the `PersistentVolumeClaim` object, which is `openshift-image-registry`.**
3. The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.

4. The size of the persistent volume claim.

b. Create the `PersistentVolumeClaim` object from the file:

   ```bash
   $ oc create -f pvc.yaml -n openshift-image-registry
   
   $ oc edit config.imageregistry.operator.openshift.io -o yaml
   
   **Example output**

   ```yaml
   storage:
   pvc:
     claim: 1
   ```

   1. Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see **Configuring the registry for vSphere**.

### 12.6.21. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   ```bash
   $ watch -n5 oc get clusteroperators
   
   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```bash
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

   ```bash
   $ oc get pods --all-namespaces
   ```

   **Example output**

   ```
   NAMESPACE                         NAME                                            READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator     openshift-apiserver-operator-85cb746d55-zqhs8   1/1     Running     1
   Running     9m
   openshift-apiserver              apiserver-67b9g                                 1/1     Running     0
   3m
   openshift-apiserver              apiserver-ljcmx                                 1/1     Running     0
   1m
   openshift-apiserver              apiserver-z25h4                                 1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8   1/1     Running     0
   Running     0
   5m
   ...
   ```

   b. View the logs for a pod that is listed in the output of the previous command by using the following command:

   ```bash
   $ oc logs <pod_name> -n <namespace>  
   ```

   Specify the pod name and namespace, as shown in the output of the previous command.

   If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.
12.6.22. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

Procedure

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

12.6.23. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

12.6.24. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.

12.7. INSTALLING A CLUSTER ON VMC IN A RESTRICTED NETWORK WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform version 4.6, you can install a cluster on VMware vSphere infrastructure that you provision in a restricted network by deploying it to VMware Cloud (VMC) on AWS.

Once you configure your VMC environment for OpenShift Container Platform deployment, you use the OpenShift Container Platform installation program from the bastion management host, co-located in the VMC environment. The installation program and control plane automates the process of deploying and managing the resources needed for the OpenShift Container Platform cluster.

12.7.1. Setting up VMC for vSphere
You can install OpenShift Container Platform on VMware Cloud (VMC) on AWS hosted vSphere clusters to enable applications to be deployed and managed both on-premise and off-premise, across the hybrid cloud.

You must configure several options in your VMC environment prior to installing OpenShift Container Platform on VMware vSphere. Ensure your VMC environment has the following prerequisites:

- Create a non-exclusive, DHCP-enabled, NSX-T network segment and subnet. Other virtual machines (VMs) can be hosted on the subnet, but at least eight IP addresses must be available for the OpenShift Container Platform deployment.

- Configure the following firewall rules:
  - An ANY:ANY firewall rule between the installation host and the software-defined data center (SDDC) management network on port 443. This allows you to upload the Red Hat Enterprise Linux CoreOS (RHCOS) OVA during deployment.
  - An HTTPS firewall rule between the OpenShift Container Platform compute network and vCenter. This connection allows OpenShift Container Platform to communicate with vCenter for provisioning and managing nodes, persistent volume claims (PVCs), and other resources.

- You must have the following information to deploy OpenShift Container Platform:
  - The OpenShift Container Platform cluster name, such as `vmc-prod-1`.
  - The base DNS name, such as `companyname.com`.
  - If not using the default, the pod network CIDR and services network CIDR must be identified, which are set by default to `10.128.0.0/14` and `172.30.0.0/16`, respectively. These CIDRs are used for pod-to-pod and pod-to-service communication and are not accessible externally; however, they must not overlap with existing subnets in your organization.
  - The following vCenter information:
    - vCenter hostname, username, and password
    - Datacenter name, such as `SDDC-Datacenter`
    - Cluster name, such as `Cluster-1`
    - Network name
    - Datastore name, such as `WorkloadDatastore`
It is recommended to move your vSphere cluster to the VMC Compute-ResourcePool resource pool after your cluster installation is finished.

- A Linux-based host deployed to VMC as a bastion.
  - The bastion host can be Red Hat Enterprise Linux (RHEL) or any another Linux-based host; it must have Internet connectivity and the ability to upload an OVA to the ESXi hosts.
  - Download and install the OpenShift CLI tools to the bastion host.
    - The openshift-install installation program
    - The OpenShift CLI (oc) tool

You cannot use the VMware NSX Container Plugin for Kubernetes (NCP), and NSX is not used as the OpenShift SDN. The version of NSX currently available with VMC is incompatible with the version of NCP certified with OpenShift Container Platform.

However, the NSX DHCP service is used for virtual machine IP management with the full-stack automated OpenShift Container Platform deployment and with nodes provisioned, either manually or automatically, by the Machine API integration with vSphere. Additionally, NSX firewall rules are created to enable access with the OpenShift Container Platform cluster and between the bastion host and the VMC vSphere hosts.

12.7.1.1. VMC Sizer tool

VMware Cloud on AWS is built on top of AWS bare metal infrastructure; this is the same bare metal infrastructure which runs AWS native services. When a VMware cloud on AWS software-defined data center (SDDC) is deployed, you consume these physical server nodes and run the VMware ESXi hypervisor in a single tenant fashion. This means the physical infrastructure is not accessible to anyone else using VMC. It is important to consider how many physical hosts you will need to host your virtual infrastructure.

To determine this, VMware provides the VMC on AWS Sizer. With this tool, you can define the resources you intend to host on VMC:

- Types of workloads
- Total number of virtual machines
- Specification information such as:
  - Storage requirements
  - vCPUs
  - vRAM
  - Overcommit ratios

With these details, the sizer tool can generate a report, based on VMware best practices, and recommend your cluster configuration and the number of hosts you will need.
12.7.2. vSphere prerequisites

- Create a registry on your mirror host and obtain the imageContentSources data for your version of OpenShift Container Platform.

**IMPORTANT**

Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- Provision block registry storage. For more information on persistent storage, see Understanding persistent storage.

- Review details about the OpenShift Container Platform installation and update processes.

- If you use a firewall and plan to use telemetry, you must configure the firewall to allow the sites that your cluster requires access to.

**NOTE**

Be sure to also review this site list if you are configuring a proxy.

12.7.3. About installations in restricted networks

In OpenShift Container Platform 4.6, you can perform an installation that does not require an active connection to the Internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service’s IAM service, require Internet access, so you might still require Internet access. Depending on your network, you might require less Internet access for an installation on bare metal hardware or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift Container Platform registry and contains the installation media. You can create this registry on a mirror host, which can access both the Internet and your closed network, or by using other methods that meet your restrictions.

**IMPORTANT**

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

12.7.3.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The ClusterVersion status includes an Unable to retrieve available updates error.
By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

### 12.7.4. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to obtain the images that are necessary to install your cluster.

You must have Internet access to:

- Access **OpenShift Cluster Manager** to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access **Quay.io** to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

**IMPORTANT**

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

### 12.7.5. VMware vSphere infrastructure requirements

You must install the OpenShift Container Platform cluster on a VMware vSphere version 6 or 7 instance that meets the requirements for the components that you use.

#### Table 12.48. Minimum supported vSphere version for VMware components

<table>
<thead>
<tr>
<th>Component</th>
<th>Minimum supported versions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>vSphere 6.5 and later with HW version 13</td>
<td>This version is the minimum version that Red Hat Enterprise Linux CoreOS (RHCOS) supports. See the Red Hat Enterprise Linux 8 supported hypervisors list.</td>
</tr>
<tr>
<td>Storage with in-tree drivers</td>
<td>vSphere 6.5 and later</td>
<td>This plug-in creates vSphere storage by using the in-tree storage drivers for vSphere included in OpenShift Container Platform.</td>
</tr>
</tbody>
</table>

If you use a vSphere version 6.5 instance, consider upgrading to 6.7U3 or 7.0 before you install OpenShift Container Platform.
IMPORTANT
You must ensure that the time on your ESXi hosts is synchronized before you install OpenShift Container Platform. See Edit Time Configuration for a Host in the VMware documentation.

12.7.6. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

12.7.6.1. Required machines

The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

NOTE
The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

IMPORTANT
To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits.

12.7.6.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in initramfs during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

12.7.6.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

12.7.6.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

12.7.7. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

Prerequisites

- Review the OpenShift Container Platform 4.x Tested Integrations page before you create the supporting infrastructure for your cluster.

Procedure

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

12.7.7.1. Networking requirements for user-provisioned infrastructure
All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

**Table 12.49. All machines to all machines**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

**Table 12.50. All machines to control plane**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

**Table 12.51. Control plane machines to control plane machines**
Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.

**IMPORTANT**

OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Load balancers
Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer.** Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:

   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.

   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**

   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

**Table 12.52. API load balancer**

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the /readyz endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>
The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the /readyz endpoint to the removal of the API server instance from the pool. Within the time frame after /readyz returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.

2. **Application Ingress load balancer**: Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.
DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 12.54. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IMPORTANT</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.</td>
</tr>
<tr>
<td>Routes</td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Master hosts</td>
<td>&lt;master&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>
Worker hosts  

Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.

**TIP**

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 12.13. Sample DNS zone database**

```
$TTL 1W
@ IN SOA ns1.example.com. root (  
  2019070700 ; serial  
  3H ; refresh (3 hours)  
  30M ; retry (30 minutes)  
  2W ; expiry (2 weeks)  
  1W ) ; minimum (1 week)  
IN NS ns1.example.com.  
IN MX 10 smtp.example.com.  
;  
ns1 IN A 192.168.1.5  
smtp IN A 192.168.1.5  
;  
helper IN A 192.168.1.5  
helper.ocp4 IN A 192.168.1.5  
;  
; The api identifies the IP of your load balancer.  
api.ocp4 IN A 192.168.1.5  
api-int.ocp4 IN A 192.168.1.5  
;  
; The wildcard also identifies the load balancer.  
*.apps.ocp4 IN A 192.168.1.5  
;  
; Create an entry for the bootstrap host.  
bootstrap.ocp4 IN A 192.168.1.96  
;  
; Create entries for the master hosts.  
master0.ocp4 IN A 192.168.1.97  
master1.ocp4 IN A 192.168.1.98  
master2.ocp4 IN A 192.168.1.99  
;  
; Create entries for the worker hosts.  
worker0.ocp4 IN A 192.168.1.11
```
The following example BIND zone file shows sample PTR records for reverse name resolution.

Example 12.14. Sample DNS zone database for reverse records

```text
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H ; refresh (3 hours)
  30M ; retry (30 minutes)
  2W ; expiry (2 weeks)
  1W ) ; minimum (1 week)
IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

12.7.8. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

NOTE

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

NOTE

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.
Procedure

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```bash
   $ ssh-keygen -t ed25519 -N "\n   -f <path>/<file_name>
   ```

   Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

   Running this command generates an SSH key that does not require a password in the location that you specified.

   **NOTE**

   If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture, do not create a key that uses the ed25519 algorithm. Instead, create a key that uses the rsa or ecdsa algorithm.

2. Start the `ssh-agent` process as a background task:

   ```bash
   $ eval "$(ssh-agent -s)"
   
   **Example output**

   Agent pid 31874

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the `ssh-agent`:

   ```bash
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.
12.7.9. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.
- Obtain the `imageContentSources` section from the output of the command to mirror the repository.
- Obtain the contents of the certificate for your mirror registry.

Procedure

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following `install-config.yaml` file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file `install-config.yaml`.

   - Unless you use a registry that RHCOS trusts by default, such as `docker.io`, you must provide the contents of the certificate for your mirror repository in the `additionalTrustBundle` section. In most cases, you must provide the certificate for your mirror.

   - You must include the `imageContentSources` section from the output of the command to mirror the repository.

3. Back up the `install-config.yaml` file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The `install-config.yaml` file is consumed during the next step of the installation process. You must back it up now.

12.7.9.1. Sample `install-config.yaml` file for VMware vSphere
You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```
apiVersion: v1
baseDomain: example.com
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: test
platform:
sphere:
  vcenter: your.vcenter.server
username: username
password: password
datacenter: datacenter
defaultDatastore: datastore
folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"

fips: false
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}'
sshKey: 'ssh-ed25519 AAAA...
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources:
  - mirrors:
    source: quay.io/openshift-release-dev/ocp-release
  - mirrors:
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2. The `controlPlane` section is a single mapping, but the `compute` section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3. Whether to enable or disable simultaneous multithreading, or `hyperthreading`. By default, simultaneous multithreading is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to `Disabled`. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.
IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Your machines must use at least 8 CPUs and 32 GB of RAM if you disable simultaneous multithreading.

4 You must set the value of the `replicas` parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not perform when you use user-provisioned infrastructure. You must manually deploy worker machines for the cluster to use before you finish installing OpenShift Container Platform.

7 The number of control plane machines that you add to the cluster. Because the cluster uses this values as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

8 The cluster name that you specified in your DNS records.

9 The fully-qualified hostname or IP address of the vCenter server.

10 The name of the user for accessing the server. This user must have at least the roles and privileges that are required for static or dynamic persistent volume provisioning in vSphere.

11 The password associated with the vSphere user.

12 The vSphere datacenter.

13 The default vSphere datastore to use.

14 Optional: For installer-provisioned infrastructure, the absolute path of an existing folder where the installation program creates the virtual machines, for example, `/<datacenter_name>/vm/<folder_name>/<subfolder_name>`. If you do not provide this value, the installation program creates a top-level folder in the datacenter virtual machine folder that is named with the infrastructure ID. If you are providing the infrastructure for the cluster, omit this parameter.

15 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the `x86_64` architecture.

16 For `<local_registry>`, specify the registry domain name, and optionally the port, that your mirror registry uses to serve content. For example `registry.example.com` or `registry.example.com:5000`. For `<credentials>`, specify the base64-encoded user name and password for your mirror registry.

17 The public portion of the default SSH key for the `core` user in Red Hat Enterprise Linux CoreOS (RHCOS).
NOTE
For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your `ssh-agent` process uses.

Provide the contents of the certificate file that you used for your mirror registry.

Provide the `imageContentSources` section from the output of the command to mirror the repository.

12.7.9.2. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

Prerequisites

- You have an existing `install-config.yaml` file.

- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the `Proxy` object’s `spec.noProxy` field to bypass the proxy if necessary.

NOTE
The `Proxy` object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the `Proxy` object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

Procedure

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

```yaml
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
  additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
```

...
1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be http.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, .y.com matches x.y.com, but not y.com. Use * to bypass the proxy for all destinations. You must include vCenter’s IP address and the IP range that you use for its machines.

4. If provided, the installation program generates a config map that is named user-ca-bundle in the openshift-config namespace to hold the additional CA certificates. If you provide additionalTrustBundle and at least one proxy setting, the Proxy object is configured to reference the user-ca-bundle config map in the trustedCA field. The Cluster Network Operator then creates a trusted-ca-bundle config map that merges the contents specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

**NOTE**

The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

**NOTE**

Only the Proxy object named cluster is supported, and no additional proxies can be created.

### 12.7.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program. For a restricted network installation, these files are on your mirror host.

- You created the `install-config.yaml` installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```sh
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Example output

   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift

   1 For `<installation_directory>`, specify the installation directory that contains the `install-config.yaml` file you created.

2. Remove the Kubernetes manifest files that define the control plane machines and compute machine sets:

   ```sh
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   Because you create and manage these resources yourself, you do not have to initialize them.

   - You can preserve the machine set files to create compute machines by using the machine API, but you must update references to them to match your environment.

3. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.

b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.

c. Save and exit the file.

4. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```bash
   $ ./openshift-install create ignition-configs --dir <installation_directory> 1
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ├── auth
   │   └── kubeadmin-password
   │       └── kubeconfig
   │           ├── bootstrap.ign
   │           └── master.ign
   │               └── metadata.json
   │                   └── worker.ign
   
12.7.11. Extracting the infrastructure name

   The Ignition config files contain a unique cluster identifier that you can use to uniquely identify your cluster in VMware Cloud on AWS. If you plan to use the cluster identifier as the name of your virtual machine folder, you must extract it.

   **Prerequisites**

   • You obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

   • You generated the Ignition config files for your cluster.

   • You installed the `jq` package.

   **Procedure**

   • To extract and view the infrastructure name from the Ignition config file metadata, run the following command:

     ```bash
     $ jq -r .infraID <installation_directory>/metadata.json 1
     ```

     For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

   **Example output**

     `openshift-vw9j6`
The output of this command is your cluster name and a random string.

12.7.12. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

Before you install a cluster that contains user-provisioned infrastructure on VMware vSphere, you must create RHCOS machines on vSphere hosts for it to use.

Prerequisites

- You have obtained the Ignition config files for your cluster.
- You have access to an HTTP server that you can access from your computer and that the machines that you create can access.
- You have created a vSphere cluster.

Procedure

1. Upload the bootstrap Ignition config file, which is named `<installation_directory>/bootstrap.ign`, that the installation program created to your HTTP server. Note the URL of this file.

2. Save the following secondary Ignition config file for your bootstrap node to your computer as `<installation_directory>/merge-bootstrap.ign`:

   ```json
   {      
      "ignition": {      
         "config": {      
            "merge": [      
               {      
                  "source": "<bootstrap_ignition_config_url>",  
                  "verification": {}      
               }      
            ]      
         },      
         "timeouts": {},      
         "version": "3.1.0"      
      },      
      "networkd": {},      
      "passwd": {},      
      "storage": {},      
      "systemd": {}      
   }
   ```

   Specify the URL of the bootstrap Ignition config file that you hosted.

   When you create the virtual machine (VM) for the bootstrap machine, you use this Ignition config file.

3. Locate the following Ignition config files that the installation program created:

   - `<installation_directory>/master.ign`
   - `<installation_directory>/worker.ign`
4. Convert the Ignition config files to Base64 encoding. Later in this procedure, you must add these files to the extra configuration parameter `guestinfo.ignition.config.data` in your VM. For example, if you use a Linux operating system, you can use the `base64` command to encode the files.

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
$ base64 -w0 <installation_directory>/merge-bootstrap.ign > <installation_directory>/merge-bootstrap.64
```

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

5. Obtain the RHCOS OVA image. Images are available from the RHCOS image mirror page.

**IMPORTANT**

The RHCOS images might not change with every release of OpenShift Container Platform. You must download an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

The filename contains the OpenShift Container Platform version number in the format `rhcos-vmware.<architecture>.ova`.

6. In the vSphere Client, create a folder in your datacenter to store your VMs.
   a. Click the VMs and Templates view.
   b. Right-click the name of your datacenter.
   c. Click New Folder → New VM and Template Folder.
   d. In the window that is displayed, enter the folder name. If you did not specify an existing folder in the `install-config.yaml` file, then create a folder with the same name as the infrastructure ID. You use this folder name so vCenter dynamically provisions storage in the appropriate location for its Workspace configuration.

7. In the vSphere Client, create a template for the OVA image and then clone the template as needed.

**NOTE**

In the following steps, you create a template and then clone the template for all of your cluster machines. You then provide the location for the Ignition config file for that cloned machine type when you provision the VMs.
a. From the Hosts and Clusters tab, right-click your cluster name and select Deploy OVF Template.

b. On the Select an OVF tab, specify the name of the RHCOS OVA file that you downloaded.

c. On the Select a name and folder tab, set a Virtual machine name for your template, such as Template-RHCOS. Click the name of your vSphere cluster and select the folder you created in the previous step.

d. On the Select a compute resource tab, click the name of your vSphere cluster.

e. On the Select storage tab, configure the storage options for your VM.
   - Select Thin Provision or Thick Provision, based on your storage preferences.
   - Select the datastore that you specified in your install-config.yaml file.

f. On the Select network tab, specify the network that you configured for the cluster, if available.

g. When creating the OVF template, do not specify values on the Customize template tab or configure the template any further.

**IMPORTANT**

Do not start the original VM template. The VM template must remain off and must be cloned for new RHCOS machines. Starting the VM template configures the VM template as a VM on the platform, which prevents it from being used as a template that machine sets can apply configurations to.

8. After the template deploys, deploy a VM for a machine in the cluster.

   a. Right-click the template name and click Clone → Clone to Virtual Machine

   b. On the Select a name and folder tab, specify a name for the VM. You might include the machine type in the name, such as control-plane-0 or compute-1.

   c. On the Select a name and folder tab, select the name of the folder that you created for the cluster.

   d. On the Select a compute resource tab, select the name of a host in your datacenter. For a bootstrap machine, specify the URL of the bootstrap Ignition config file that you hosted.

   e. Optional: On the Select storage tab, customize the storage options.

   f. On the Select clone options, select Customize this virtual machine’s hardware

   g. On the Customize hardware tab, click VM Options → Advanced.

   - Optional: Override default DHCP networking in vSphere. To enable static IP networking:

     i. Set your static IP configuration:

     ```
     $ export IPCFG="ip=<ip>::<gateway>::<netmask>::<hostname>::<iface>:none
     nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
     ```
Example command

$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"

ii. Set the **guestinfo.afterburn.initrd.network-kargs** property before booting a VM from an OVA in vSphere:

$ govc vm.change -vm "<vm_name>" -e "guestinfo.afterburn.initrd.network-kargs=${IPCFG}"

- Optional: In the event of cluster performance issues, from the **Latency Sensitivity** list, select **High**. Ensure that your VM’s CPU and memory reservation have the following values:
  - Memory reservation value must be equal to its configured memory size.
  - CPU reservation value must be at least the number of low latency virtual CPUs multiplied by the measured physical CPU speed.

- Click **Edit Configuration**, and on the **Configuration Parameters** window, click **Add Configuration Params**. Define the following parameter names and values:
  - **guestinfo.ignition.config.data**: Locate the base-64 encoded files that you created previously in this procedure, and paste the contents of the base64-encoded Ignition config file for this machine type.
  - **guestinfo.ignition.config.data.encoding**: Specify **base64**.
  - **disk.EnableUUID**: Specify **TRUE**.

h. In the **Virtual Hardware** panel of the **Customize hardware** tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type.

i. Complete the configuration and power on the VM.

9. Create the rest of the machines for your cluster by following the preceding steps for each machine.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. Because some pods are deployed on compute machines by default, also create at least two compute machines before you install the cluster.

12.7.13. Creating more Red Hat Enterprise Linux CoreOS (RHCOS) machines in vSphere

You can create more compute machines for your cluster that uses user-provisioned infrastructure on VMware vSphere.

**Prerequisites**

- Obtain the base64-encoded Ignition file for your compute machines.
• You have access to the vSphere template that you created for your cluster.

Procedure

1. After the template deploys, deploy a VM for a machine in the cluster.
   a. Right-click the template’s name and click Clone → Clone to Virtual Machine
   b. On the Select a name and folder tab, specify a name for the VM. You might include the machine type in the name, such as compute-1.
   c. On the Select a name and folder tab, select the name of the folder that you created for the cluster.
   d. On the Select a compute resource tab, select the name of a host in your datacenter.
   e. Optional: On the Select storage tab, customize the storage options.
   f. On the Select clone options, select Customize this virtual machine’s hardware
   g. On the Customize hardware tab, click VM Options → Advanced.
      • From the Latency Sensitivity list, select High.
      • Click Edit Configuration, and on the Configuration Parameters window, click Add Configuration Params. Define the following parameter names and values:
         ○ guestinfo.ignition.config.data: Paste the contents of the base64-encoded compute Ignition config file for this machine type.
         ○ guestinfo.ignition.config.data.encoding: Specify base64.
         ○ disk.EnableUUID: Specify TRUE.
   h. In the Virtual Hardware panel of the Customize hardware tab, modify the specified values as required. Ensure that the amount of RAM, CPU, and disk storage meets the minimum requirements for the machine type. Also, make sure to select the correct network under Add network adapter if there are multiple networks available.
   i. Complete the configuration and power on the VM.

2. Continue to create more compute machines for your cluster.

12.7.14. Disk partitioning

In most cases, data partitions are originally created by installing RHCOS, rather than by installing another operating system. In such cases, the OpenShift Container Platform installer should be allowed to configure your disk partitions.

However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

• Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making /var or a subdirectory of /var, such as /var/lib/etcd, a separate partition, but not both.
Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

Creating a separate `/var` partition
In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.
- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:
   
   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:
   
   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ? SSH Public Key ...
   $ ls $HOME/clusterconfig/openshift/
   99_kubeadmin-password-secret.yaml
   99_openshift-cluster-api_master-machines-0.yaml
   99_openshift-cluster-api_master-machines-1.yaml
   99_openshift-cluster-api_master-machines-2.yaml
   ...
   ```

3. Create a `MachineConfig` object and add it to a file in the `openshift` directory. For example, name the file `98-var-partition.yaml`, change the disk device name to the name of the storage...
device on the **worker** systems, and set the storage size as appropriate. This example places the 
/var directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
  storage:
    disks:
      - device: /dev/<device_name>  
        partitions:
          - label: var
            startMiB: <partition_start_offset>  
            sizeMiB: <partition_size>  
    filesystems:
      - device: /dev/disk/by-partlabel/var
        path: /var
        format: xfs
  systemd:
    units:
      - name: var.mount  
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota
          [Install]
          WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
3. The size of the data partition in mebibytes.
4. The name of the mount unit must match the directory specified in the **Where** directive. For example, for a filesystem mounted on /var/lib/containers, the unit must be named **var-lib-containers.mount**.
5. The **prjquota** mount option must be enabled for filesystems used for container storage.
NOTE

When creating a separate /var partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run openshift-install again to create Ignition configs from a set of files in the manifest and openshift subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
```

```
$ ls $HOME/clusterconfig/
auth bootstrap.ign master.ign metadata.json worker.ign
```

Now you can use the Ignition config files as input to the vSphere installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

12.7.15. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
- You obtained the installation program and generated the Ignition config files for your cluster.
- You used the Ignition config files to create RHCOS machines for your cluster.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
--log-level=info 2
```

1. For <installation_directory>, specify the path to the directory that you stored the installation files in.

2. To view different installation details, specify warn, debug, or error instead of info.

Example output

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.
12.7.16. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

**Prerequisites**

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

**Procedure**

1. Export the `kubeadmin` credentials:

   ```bash
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig
   ```
   
   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   ```bash
   $ oc whoami
   ```

   **Example output**
   
   `system:admin`

12.7.17. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

**Prerequisites**

- You added machines to your cluster.

**Procedure**

1. Confirm that the cluster recognizes the machines:

   ```bash
   $ oc get nodes
   ```

   **Example output**
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0

The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   ```shell
   $ oc get csr
   
   Example output
   
   NAME      AGE  REQUESTOR                                                                 CONDITION
   csr-8b2br  15m  system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   csr-8vnps  15m  system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
   ...
   
   In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

   NOTE

   Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the `machine-approver` if the Kubelet requests a new certificate with identical parameters.
NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the `oc exec`, `oc rsh`, and `oc logs` commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the `node-bootstrapper` service account in the `system:node` or `system:admin` groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name> ①
  ``

  ① `<csr_name>` is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{
  is the name of a CSR from the list of current CSRs.

        }}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ``

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```bash
$ oc get csr
```

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>REQUESTOR</th>
<th>CONDITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>csr-bfd72</td>
<td>5m26s</td>
<td>system:node:ip-10-0-50-126.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>csr-c57t</td>
<td>5m26s</td>
<td>system:node:ip-10-0-95-157.us-east-2.compute.internal</td>
<td>Pending</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name> ①
  ``

  ① `<csr_name>` is the name of a CSR from the list of current CSRs.
To approve all pending CSRs, run the following command:

```bash
$ oc get csr -o go-template="{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}" | xargs oc adm certificate approve
```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

```bash
$ oc get nodes
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

**NOTE**

It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see **Certificate Signing Requests**.

### 12.7.18. Initial Operator configuration

After the control plane initializes, you must immediately configure some Operators so that they all become available.

**Prerequisites**

- Your control plane has initialized.

**Procedure**

1. Watch the cluster components come online:

```bash
$ watch -n5 oc get clusteroperators
```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
</tbody>
</table>
12.7.18.1. Disabling the default OperatorHub sources

Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

Procedure

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```bash
  $ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
  ```

TIP

Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

12.7.18.2. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.
Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

12.7.18.2.1. Configuring registry storage for VMware vSphere

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**

- Cluster administrator permissions.
- A cluster on VMware vSphere.
- Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

- Must have "100Gi" capacity.

**IMPORTANT**

Testing shows issues with using the NFS server on RHEL as storage backend for core services. This includes the OpenShift Container Registry and Quay, Prometheus for monitoring storage, and Elasticsearch for logging storage. Therefore, using RHEL NFS to back PVs used by core services is not recommended.

Other NFS implementations on the marketplace might not have these issues. Contact the individual NFS implementation vendor for more information on any testing that was possibly completed against these OpenShift Container Platform core components.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**
   
   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```
   $ oc get pod -n openshift-image-registry
   ```
NOTE

If the storage type is **emptyDIR**, the replica number cannot be greater than **1**.

3. Check the registry configuration:

   ```
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```
   storage:
   pvc:
   claim: 1
   ```

   Leave the **claim** field blank to allow the automatic creation of an **image-registry-storage PVC**.

4. Check the **clusteroperator** status:

   ```
   $ oc get clusteroperator image-registry
   ```

### 12.7.18.2.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

  ```
  $ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {
  "storage": {
    "emptyDir": {}}}}'
  ```

  **WARNING**

  Configure this option for only non-production clusters.

  If you run this command before the Image Registry Operator initializes its components, the **oc patch** command fails with the following error:

  ```
  Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
  ```

  Wait a few minutes and run the command again.

### 12.7.18.2.3. Configuring block registry storage for VMware vSphere
To allow the image registry to use block storage types such as vSphere Virtual Machine Disk (VMDK) during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.

**Procedure**

1. To set the image registry storage as a block storage type, patch the registry so that it uses the **Recreate** rollout strategy and runs with only 1 replica:

   ```bash
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
    "rolloutStrategy":"Recreate","replicas":1}'}
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.
   
   a. Create a `pvc.yaml` file with the following contents to define a VMware vSphere **PersistentVolumeClaim** object:

   ```yaml
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
   name: image-registry-storage
   namespace: openshift-image-registry
   spec:
   accessModes:
   - ReadWriteOnce
   resources:
   requests:
   storage: 100Gi
   ```

   1. A unique name that represents the **PersistentVolumeClaim** object.
   2. The namespace for the **PersistentVolumeClaim** object, which is **openshift-image-registry**.
   3. The access mode of the persistent volume claim. With **ReadWriteOnce**, the volume can be mounted with read and write permissions by a single node.
   4. The size of the persistent volume claim.

   b. Create the **PersistentVolumeClaim** object from the file:

   ```bash
   $ oc create -f pvc.yaml -n openshift-image-registry
   ```

3. Edit the registry configuration so that it references the correct PVC:

   ```bash
   $ oc edit config.imageregistry.operator.openshift.io -o yaml
   ```
Creating a custom PVC allows you to leave the `claim` field blank for the default automatic creation of an `image-registry-storage` PVC.

For instructions about configuring registry storage so that it references the correct PVC, see [Configuring registry storage for VMware vSphere](#).

### 12.7.19. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   ```bash
   $ watch -n5 oc get clusteroperators
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>6h31m</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

### Example output

INFO Waiting up to 30m0s for the cluster to initialize...

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

   a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

### Example output

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>network</td>
<td>4.6.0 True False False 29h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0 True False False 4h30m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0 True False False 3h56m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0 True False False 4h36m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0 True False False 4h36m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0 True False False 29h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0 True False False 29h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0 True False False 3h59m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0 True False False 29h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0 True False False 4h30m</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace>
```

Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

3. Register your cluster on the Cluster registration page.

You can add extra compute machines after the cluster installation is completed by following Adding compute machines to vSphere.

### 12.7.20. Backing up VMware vSphere volumes

OpenShift Container Platform provisions new volumes as independent persistent disks to freely attach and detach the volume on any node in the cluster. As a consequence, it is not possible to back up volumes that use snapshots, or to restore volumes from snapshots. See Snapshot Limitations for more information.

**Procedure**

To create a backup of persistent volumes:

1. Stop the application that is using the persistent volume.
2. Clone the persistent volume.
3. Restart the application.
4. Create a backup of the cloned volume.
5. Delete the cloned volume.

### 12.7.21. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.
After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

12.7.22. Next steps

- Customize your cluster.
- Configure image streams for the Cluster Samples Operator and the must-gather tool.
- Learn how to use Operator Lifecycle Manager (OLM) on restricted networks.
- If the mirror registry that you used to install your cluster has a trusted CA, add it to the cluster by configuring additional trust stores.
- If necessary, you can opt out of remote health reporting.

12.8. UNINSTALLING A CLUSTER ON VMC

You can remove a cluster installed on VMware vSphere infrastructure that you deployed to VMware Cloud (VMC) on AWS by using installer-provisioned infrastructure.

12.8.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

- Have a copy of the installation program that you used to deploy the cluster.
- Have the files that the installation program generated when you created your cluster.

Procedure

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

   ```
   $ ./openshift-install destroy cluster \
   --dir <installation_directory> --log-level info
   ```

   1 For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   2 To view different details, specify warn, debug, or error instead of info.
NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the `metadata.json` file in this directory to delete the cluster.

2. Optional: Delete the `<installation_directory>` directory and the OpenShift Container Platform installation program.
CHAPTER 13. INSTALLING ON ANY PLATFORM

13.1. INSTALLING A CLUSTER ON ANY PLATFORM

In OpenShift Container Platform version 4.6, you can install a cluster on any infrastructure that you provision, including virtualization and cloud environments.

IMPORTANT

Review the information in the guidelines for deploying OpenShift Container Platform on non-tested platforms before you attempt to install an OpenShift Container Platform cluster in virtualized or cloud environments.

13.1.1. Prerequisites

- Review details about the OpenShift Container Platform installation and update processes.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

NOTE

Be sure to also review this site list if you are configuring a proxy.

13.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.6, you require access to the Internet to install your cluster.

You must have Internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct Internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the content that is required and use it to populate a mirror registry with the packages that you need to install a cluster and generate the installation program. With some installation types, the environment that you install your cluster in will not require Internet access. Before you update the cluster, you update the content of the mirror registry.

13.1.3. Machine requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

13.1.3.1. Required machines
The smallest OpenShift Container Platform clusters require the following hosts:

- One temporary bootstrap machine
- Three control plane, or master, machines
- At least two compute machines, which are also known as worker machines.

**NOTE**

The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.

**IMPORTANT**

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7.9.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 8 and inherits all of its hardware certifications and requirements. See [Red Hat Enterprise Linux technology capabilities and limits](#).

### 13.1.3.2. Network connectivity requirements

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in *initramfs* during boot to fetch Ignition config files from the Machine Config Server. During the initial boot, the machines require either a DHCP server or that static IP addresses be set in order to establish a network connection to download their Ignition config files. Additionally, each OpenShift Container Platform node in the cluster must have access to a Network Time Protocol (NTP) server. If a DHCP server provides NTP servers information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

### 13.1.3.3. Minimum resource requirements

Each cluster machine must meet the following minimum requirements:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootstrap</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Control plane</td>
<td>RHCOS</td>
<td>4</td>
<td>16 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
<tr>
<td>Compute</td>
<td>RHCOS or RHEL 7.9</td>
<td>2</td>
<td>8 GB</td>
<td>100 GB</td>
<td>300</td>
</tr>
</tbody>
</table>
1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: \((\text{threads per core} \times \text{cores}) \times \text{sockets} = \text{vCPUs}\).

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

### 13.1.3.4. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The `kube-controller-manager` only approves the kubelet client CSRs. The `machine-approver` cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

### 13.1.4. Creating the user-provisioned infrastructure

Before you deploy an OpenShift Container Platform cluster that uses user-provisioned infrastructure, you must create the underlying infrastructure.

**Prerequisites**

- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

**Procedure**

1. Configure DHCP or set static IP addresses on each node.
2. Provision the required load balancers.
3. Configure the ports for your machines.
4. Configure DNS.
5. Ensure network connectivity.

### 13.1.4.1. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require network in **initramfs** during boot to fetch Ignition config from the machine config server.

During the initial boot, the machines require either a DHCP server or that static IP addresses be set on each host in the cluster in order to establish a network connection, which allows them to download their Ignition config files.

It is recommended to use the DHCP server to manage the machines for the cluster long-term. Ensure that the DHCP server is configured to provide persistent IP addresses and host names to the cluster machines.

The Kubernetes API server must be able to resolve the node names of the cluster machines. If the API
servers and worker nodes are in different zones, you can configure a default DNS search zone to allow the API server to resolve the node names. Another supported approach is to always refer to hosts by their fully-qualified domain names in both the node objects and all DNS requests.

You must configure the network connectivity between machines to allow cluster components to communicate. Each machine must be able to resolve the host names of all other machines in the cluster.

Table 13.1. All machines to all machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP</td>
<td>N/A</td>
<td>Network reachability tests</td>
</tr>
<tr>
<td>TCP</td>
<td>1936</td>
<td>Metrics</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099.</td>
</tr>
<tr>
<td></td>
<td>10250-10259</td>
<td>The default ports that Kubernetes reserves</td>
</tr>
<tr>
<td></td>
<td>10256</td>
<td>openshift-sdn</td>
</tr>
<tr>
<td>UDP</td>
<td>4789</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>6081</td>
<td>VXLAN and Geneve</td>
</tr>
<tr>
<td></td>
<td>9000-9999</td>
<td>Host level services, including the node exporter on ports 9100-9101.</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>30000-32767</td>
<td>Kubernetes node port</td>
</tr>
</tbody>
</table>

Table 13.2. All machines to control plane

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>6443</td>
<td>Kubernetes API</td>
</tr>
</tbody>
</table>

Table 13.3. Control plane machines to control plane machines

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>2379-2380</td>
<td>etcd server and peer ports</td>
</tr>
</tbody>
</table>

Network topology requirements
The infrastructure that you provision for your cluster must meet the following network topology requirements.
OpenShift Container Platform requires all nodes to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

### Load balancers

Before you install OpenShift Container Platform, you must provision two load balancers that meet the following requirements:

1. **API load balancer**: Provides a common endpoint for users, both human and machine, to interact with and configure the platform. Configure the following conditions:
   - Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the API routes.
   - A stateless load balancing algorithm. The options vary based on the load balancer implementation.

   **IMPORTANT**
   Do not configure session persistence for an API load balancer.

Configure the following ports on both the front and back of the load balancers:

#### Table 13.4. API load balancer

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6443</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane. You must configure the <code>/readyz</code> endpoint for the API server health check probe.</td>
<td>X</td>
<td>X</td>
<td>Kubernetes API server</td>
</tr>
<tr>
<td>22623</td>
<td>Bootstrap and control plane. You remove the bootstrap machine from the load balancer after the bootstrap machine initializes the cluster control plane.</td>
<td>X</td>
<td></td>
<td>Machine config server</td>
</tr>
</tbody>
</table>

**NOTE**

The load balancer must be configured to take a maximum of 30 seconds from the time the API server turns off the `/readyz` endpoint to the removal of the API server instance from the pool. Within the time frame after `/readyz` returns an error or becomes healthy, the endpoint must have been removed or added. Probing every 5 or 10 seconds, with two successful requests to become healthy and three to become unhealthy, are well-tested values.
2. **Application Ingress load balancer.** Provides an Ingress point for application traffic flowing in from outside the cluster. Configure the following conditions:

- Layer 4 load balancing only. This can be referred to as Raw TCP, SSL Passthrough, or SSL Bridge mode. If you use SSL Bridge mode, you must enable Server Name Indication (SNI) for the Ingress routes.

- A connection-based or session-based persistence is recommended, based on the options available and types of applications that will be hosted on the platform.

Configure the following ports on both the front and back of the load balancers:

<table>
<thead>
<tr>
<th>Port</th>
<th>Back-end machines (pool members)</th>
<th>Internal</th>
<th>External</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>443</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTPS traffic</td>
</tr>
<tr>
<td>80</td>
<td>The machines that run the Ingress router pods, compute, or worker, by default.</td>
<td>X</td>
<td>X</td>
<td>HTTP traffic</td>
</tr>
</tbody>
</table>

**TIP**

If the true IP address of the client can be seen by the load balancer, enabling source IP-based session persistence can improve performance for applications that use end-to-end TLS encryption.

**NOTE**

A working configuration for the Ingress router is required for an OpenShift Container Platform cluster. You must configure the Ingress router after the control plane initializes.

**NTP configuration**

OpenShift Container Platform clusters are configured to use a public Network Time Protocol (NTP) server by default. If you want to use a local enterprise NTP server, or if your cluster is being deployed in a disconnected network, you can configure the cluster to use a specific time server. For more information, see the documentation for Configuring chrony time service.

If a DHCP server provides NTP server information, the chrony time service on the Red Hat Enterprise Linux CoreOS (RHCOS) machines read the information and can sync the clock with the NTP servers.

**Additional resources**

- Configuring chrony time service

**13.1.4.2. User-provisioned DNS requirements**

DNS is used for name resolution and reverse name resolution. DNS A/AAAA or CNAME records are used for name resolution and PTR records are used for reverse name resolution. The reverse records are important because Red Hat Enterprise Linux CoreOS (RHCOS) uses the reverse records to set the
host name for all the nodes. Additionally, the reverse records are used to generate the certificate signing requests (CSR) that OpenShift Container Platform needs to operate.

The following DNS records are required for an OpenShift Container Platform cluster that uses user-provisioned infrastructure. In each record, `<cluster_name>` is the cluster name and `<base_domain>` is the cluster base domain that you specify in the `install-config.yaml` file. A complete DNS record takes the form: `<component>.<cluster_name>.<base_domain>`.

Table 13.6. Required DNS records

<table>
<thead>
<tr>
<th>Component</th>
<th>Record</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes API</td>
<td>api.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>api-int.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the load balancer for the control plane machines. These records must be resolvable from all the nodes within the cluster.</td>
</tr>
<tr>
<td></td>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a wildcard DNS A/AAAA or CNAME record that refers to the load balancer that targets the machines that run the Ingress router pods, which are the worker nodes by default. These records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.</td>
</tr>
<tr>
<td>Bootstrap</td>
<td>bootstrap.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add a DNS A/AAAA or CNAME record, and a DNS PTR record, to identify the bootstrap machine. These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Master hosts</td>
<td>&lt;master&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the control plane nodes (also known as the master nodes). These records must be resolvable by the nodes within the cluster.</td>
</tr>
<tr>
<td>Worker hosts</td>
<td>&lt;worker&gt;&lt;n&gt;.&lt;cluster_name&gt;.&lt;base_domain&gt;.</td>
<td>Add DNS A/AAAA or CNAME records and DNS PTR records to identify each machine for the worker nodes. These records must be resolvable by the nodes within the cluster.</td>
</tr>
</tbody>
</table>

**IMPORTANT**

The API server must be able to resolve the worker nodes by the host names that are recorded in Kubernetes. If the API server cannot resolve the node names, then proxied API calls can fail, and you cannot retrieve logs from pods.
TIP

You can use the `nslookup <hostname>` command to verify name resolution. You can use the `dig -x <ip_address>` command to verify reverse name resolution for the PTR records.

The following example of a BIND zone file shows sample A records for name resolution. The purpose of the example is to show the records that are needed. The example is not meant to provide advice for choosing one name resolution service over another.

**Example 13.1. Sample DNS zone database**

```
$TTL 1W
@ IN SOA ns1.example.com. root (2019070700 ; serial 3H ; refresh (3 hours) 30M ; retry (30 minutes) 2W ; expiry (2 weeks) 1W ) ; minimum (1 week)
IN NS ns1.example.com.
IN MX 10 smtp.example.com.

ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5

helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5

api.ocp4 IN A 192.168.1.5
api-int.ocp4 IN A 192.168.1.5

* apps.ocp4 IN A 192.168.1.5

bootstrap.ocp4 IN A 192.168.1.96

master0.ocp4 IN A 192.168.1.97
master1.ocp4 IN A 192.168.1.98
master2.ocp4 IN A 192.168.1.99

worker0.ocp4 IN A 192.168.1.11
worker1.ocp4 IN A 192.168.1.7

;EOF
```

The following example BIND zone file shows sample PTR records for reverse name resolution.

**Example 13.2. Sample DNS zone database for reverse records**

```
```
13.1.5. Generating an SSH private key and adding it to the agent

If you want to perform installation debugging or disaster recovery on your cluster, you must provide an SSH key to both your ssh-agent and the installation program. You can use this key to access the bootstrap machine in a public cluster to troubleshoot installation issues.

**NOTE**

In a production environment, you require disaster recovery and debugging.

You can use this key to SSH into the master nodes as the user core. When you deploy the cluster, the key is added to the core user’s ~/.ssh/authorized_keys list.

**NOTE**

You must use a local key, not one that you configured with platform-specific approaches such as AWS key pairs.

**Procedure**

1. If you do not have an SSH key that is configured for password-less authentication on your computer, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N "" \
-f <path>/<file_name>
```
Specify the path and file name, such as `~/.ssh/id_rsa`, of the new SSH key. If you have an existing key pair, ensure your public key is in the your `~/.ssh` directory.

Running this command generates an SSH key that does not require a password in the location that you specified.

**NOTE**

If you plan to install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the `x86_64` architecture, do not create a key that uses the `ed25519` algorithm. Instead, create a key that uses the `rsa` or `ecdsa` algorithm.

2. Start the **ssh-agent** process as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   
   Example output
   
   Agent pid 31874
   ```

   **NOTE**

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

3. Add your SSH private key to the **ssh-agent**:

   ```
   $ ssh-add <path>/<file_name>
   ```

   **Example output**

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

   Specify the path and file name for your SSH private key, such as `~/.ssh/id_rsa`

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide this key to your cluster’s machines.

**13.1.6. Obtaining the installation program**

Before you install OpenShift Container Platform, download the installation file on a local computer.

**Prerequisites**

- You have a computer that runs Linux or macOS, with 500 MB of local disk space
Procedure

1. Access the Infrastructure Provider page on the OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

2. Select your infrastructure provider.

3. Navigate to the page for your installation type, download the installation program for your operating system, and place the file in the directory where you will store the installation configuration files.

   **IMPORTANT**
   
   The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both files are required to delete the cluster.

   **IMPORTANT**
   
   Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

4. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   $ tar xvf openshift-install-linux.tar.gz

5. Download your installation pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

13.1.7. Installing the OpenShift CLI by downloading the binary

You can install the OpenShift CLI (oc) in order to interact with OpenShift Container Platform from a command-line interface. You can install oc on Linux, Windows, or macOS.

   **IMPORTANT**
   
   If you installed an earlier version of oc, you cannot use it to complete all of the commands in OpenShift Container Platform 4.6. Download and install the new version of oc.

13.1.7.1. Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (oc) binary on Linux by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Linux Client entry and save the file.

4. Unpack the archive:

   \$ tar xvzf <file>

5. Place the oc binary in a directory that is on your PATH.
   
   To check your PATH, execute the following command:

   \$ echo $PATH

After you install the OpenShift CLI, it is available using the oc command:

\$ oc <command>

13.1.7.2. Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (oc) binary on Windows by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 Windows Client entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the oc binary to a directory that is on your PATH.
   
   To check your PATH, open the command prompt and execute the following command:

   C:\> path

After you install the OpenShift CLI, it is available using the oc command:

C:\> oc <command>

13.1.7.3. Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (oc) binary on macOS by using the following procedure.

Procedure


2. Select the appropriate version in the Version drop-down menu.

3. Click Download Now next to the OpenShift v4.6 MacOSX Client entry and save the file.
4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

   After you install the OpenShift CLI, it is available using the **oc** command:

   ```
   $ oc <command>
   ```

### 13.1.8. Manually creating the installation configuration file

For installations of OpenShift Container Platform that use user-provisioned infrastructure, you manually generate your installation configuration file.

**Prerequisites**

- Obtain the OpenShift Container Platform installation program and the access token for your cluster.

**Procedure**

1. Create an installation directory to store your required installation assets in:

   ```
   $ mkdir <installation_directory>
   ```

   **IMPORTANT**

   You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the following **install-config.yaml** file template and save it in the `<installation_directory>`.

   **NOTE**

   You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

   **IMPORTANT**

   The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.
You can customize the `install-config.yaml` file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

```yaml
apiVersion: v1
baseDomain: example.com  # 1
compute:  # 2
  - hyperthreading: Enabled  # 3
    name: worker
    replicas: 0  # 4
controlPlane:  # 5
  hyperthreading: Enabled  # 6
  name: master
  replicas: 3  # 7
metadata:  # 8
  name: test
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14  # 9
  hostPrefix: 23  # 10
  networkType: OpenShiftSDN
  serviceNetwork:  # 11
    - 172.30.0.0/16
platform:
  none: {}  # 12
fips: false  # 13
pullSecret: '{"auths": ...}'  # 14
sshKey: 'ssh-ed25519 AAAA...'  # 15
```

1. The base domain of the cluster. All DNS records must be sub-domains of this base and include the cluster name.

2-5. The `controlPlane` section is a single mapping, but the `compute` section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the `compute` section must begin with a hyphen, `-`, and the first line of the `controlPlane` section must not. Although both sections currently define a single machine pool, it is possible that future versions of OpenShift Container Platform will support defining multiple compute pools during installation. Only one control plane pool is used.

3-6. Whether to enable or disable simultaneous multithreading (SMT), or `hyperthreading`. By default, SMT is enabled to increase the performance of your machines’ cores. You can disable it by setting the parameter value to `Disabled`. If you disable SMT, you must disable it in all cluster machines; this includes both control plane and compute machines.

**NOTE**

Simultaneous multithreading (SMT) is enabled by default. If SMT is not enabled in your BIOS settings, the `hyperthreading` parameter has no effect.

**IMPORTANT**

If you disable `hyperthreading`, whether in the BIOS or in the `install-config.yaml`, ensure that your capacity planning accounts for the dramatically decreased machine performance.
You must set the value of the `replicas` parameter to 0. This parameter controls the number of workers that the cluster creates and manages for you, which are functions that the cluster does not control plane machines that you add to the cluster. Because the cluster uses this number as the number of etcd endpoints in the cluster, the value must match the number of control plane machines that you deploy.

The cluster name that you specified in your DNS records.

A block of IP addresses from which pod IP addresses are allocated. This block must not overlap with existing physical networks. These IP addresses are used for the pod network. If you need to access the pods from an external network, you must configure load balancers and routers to manage the traffic.

The subnet prefix length to assign to each individual node. For example, if `hostPrefix` is set to 23, then each node is assigned a /23 subnet out of the given `cidr`, which allows for 510 \((2^{32} - 23)\) pod IP addresses. If you are required to provide access to nodes from an external network, configure load balancers and routers to manage the traffic.

The IP address pool to use for service IP addresses. You can enter only one IP address pool. This block must not overlap with existing physical networks. If you need to access the services from an external network, configure load balancers and routers to manage the traffic.

You must set the platform to `none`. You cannot provide additional platform configuration variables for your platform.

Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

**IMPORTANT**

The use of FIPS Validated / Modules in Process cryptographic libraries is only supported on OpenShift Container Platform deployments on the x86_64 architecture.

The pull secret from the Red Hat OpenShift Cluster Manager. This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

The public portion of the default SSH key for the core user in Red Hat Enterprise Linux CoreOS (RHCOS).

**NOTE**

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your ssh-agent process uses.

13.1.8.1. Configuring the cluster-wide proxy during installation
Production environments can deny direct access to the Internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the `install-config.yaml` file.

**Prerequisites**

- You have an existing `install-config.yaml` file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the Proxy object’s `spec.noProxy` field to bypass the proxy if necessary.

**NOTE**

The Proxy object `status.noProxy` field is populated with the values of the `networking.machineNetwork[].cidr`, `networking.clusterNetwork[].cidr`, and `networking.serviceNetwork[]` fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the Proxy object `status.noProxy` field is also populated with the instance metadata endpoint (169.254.169.254).

**Procedure**

1. Edit your `install-config.yaml` file and add the proxy settings. For example:

   ```yaml
   apiVersion: v1
   baseDomain: my.domain.com
   proxy:
     httpProxy: http://<username>:<pswd>@<ip>:<port>
     httpsProxy: https://<username>:<pswd>@<ip>:<port>
     noProxy: example.com
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
     <MY_TRUSTED_CA_CERT>
     -----END CERTIFICATE-----
     ...
   
   1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be `http`.
   2 A proxy URL to use for creating HTTPS connections outside the cluster.
   3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with . to match subdomains only. For example, `.y.com` matches `x.y.com`, but not `y.com`. Use * to bypass the proxy for all destinations.
   4 If provided, the installation program generates a config map that is named `user-ca-bundle` in the `openshift-config` namespace to hold the additional CA certificates. If you provide `additionalTrustBundle` and at least one proxy setting, the Proxy object is configured to reference the `user-ca-bundle` config map in the `trustedCA` field. The Cluster Network Operator then creates a `trusted-ca-bundle` config map that merges the contents
specified for the trustedCA parameter with the RHCOS trust bundle. The additionalTrustBundle field is required unless the proxy’s identity certificate is signed by an authority from the RHCOS trust bundle.

NOTE
The installation program does not support the proxy readinessEndpoints field.

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named cluster that uses the proxy settings in the provided install-config.yaml file. If no proxy settings are provided, a cluster Proxy object is still created, but it will have a nil spec.

NOTE
Only the Proxy object named cluster is supported, and no additional proxies can be created.

13.1.9. Configuring a three-node cluster

You can optionally install and run three-node clusters in OpenShift Container Platform with no workers. This provides smaller, more resource efficient clusters for cluster administrators and developers to use for development, production, and testing.

Procedure

- Edit the install-config.yaml file to set the number of compute replicas, which are also known as worker replicas, to 0, as shown in the following compute stanza:

  ```yaml
  compute:
    - name: worker
      platform: {}
      replicas: 0
  ```

13.1.10. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to make its machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to create the cluster.
IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending node-bootstrapper certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.

- You created the install-config.yaml installation configuration file.

Procedure

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   $ ./openshift-install create manifests --dir <installation_directory>

Example output

   INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
   INFO Consuming Install Config from target directory
   INFO Manifests created in: install_dir/manifests and install_dir/openshift

   1 For <installation_directory>, specify the installation directory that contains the install-config.yaml file you created.

WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.
1. Check that the `mastersSchedulable` parameter in the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` Kubernetes manifest file is set to `false`. This setting prevents pods from being scheduled on the control plane machines:
   a. Open the `<installation_directory>/manifests/cluster-scheduler-02-config.yml` file.
   b. Locate the `mastersSchedulable` parameter and ensure that it is set to `false`.
   c. Save and exit the file.

2. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```
   $ ./openshift-install create ignition-configs --dir <installation_directory>  
   ```

   For `<installation_directory>`, specify the same installation directory.

   The following files are generated in the directory:

   ```
   ├── auth
   │    └── kubeadmin-password
   │    └── kubeconfig
   │    └── bootstrap.ign
   │    └── master.ign
   │    └── metadata.json
   │    └── worker.ign
   ```

### 13.1.11. Installing RHCOS and starting the OpenShift Container Platform bootstrap process

To install OpenShift Container Platform on bare metal infrastructure that you provision, you must install Red Hat Enterprise Linux CoreOS (RHCOS) on the machines. When you install RHCOS, you must provide the Ignition config file that was generated by the OpenShift Container Platform installation program for the type of machine you are installing. If you have configured suitable networking, DNS, and load balancing infrastructure, the OpenShift Container Platform bootstrap process begins automatically after the RHCOS machines have rebooted.

To install RHCOS on the machines, follow either the steps to use an ISO image or network PXE booting.
The compute node deployment steps included in this installation document are RHCOS-specific. If you choose instead to deploy RHEL-based compute nodes, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and planned for removal in a future release of OpenShift Container Platform 4.

You can configure RHCOS during ISO and PXE installations by using the following methods:

- **Kernel arguments**: You can use kernel arguments to provide installation-specific information. For example, you can specify the locations of the RHCOS installation files that you uploaded to your HTTP server and the location of the Ignition config file for the type of node you are installing. For a PXE installation, you can use the `APPEND` parameter to pass the arguments to the kernel of the live installer. For an ISO installation, you can interrupt the live installation boot process to add the kernel arguments. In both installation cases, you can use special `coreos.inst.*` arguments to direct the live installer, as well as standard installation boot arguments for turning standard kernel services on or off.

- **Ignition configs**: OpenShift Container Platform Ignition config files (*.ign) are specific to the type of node you are installing. You pass the location of a bootstrap, control plane, or compute node Ignition config file during the RHCOS installation so that it takes effect on first boot. In special cases, you can create a separate, limited Ignition config to pass to the live system. That Ignition config could do a certain set of tasks, such as reporting success to a provisioning system after completing installation. This special Ignition config is consumed by the `coreos-installer` to be applied on first boot of the installed system. Do not provide the standard control plane and compute node Ignition configs to the live ISO directly.

- **`coreos-installer`**: You can boot the live ISO installer to a shell prompt, which allows you to prepare the permanent system in a variety of ways before first boot. In particular, you can run the `coreos-installer` command to identify various artifacts to include, work with disk partitions, and set up networking. In some cases, you can configure features on the live system and copy them to the installed system.

Whether to use an ISO or PXE install depends on your situation. A PXE install requires an available DHCP service and more preparation, but can make the installation process more automated. An ISO install is a more manual process and can be inconvenient if you are setting up more than a few machines.

As of OpenShift Container Platform 4.6, the RHCOS ISO and other installation artifacts provide support for installation on disks with 4K sectors.

13.1.11.1. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines using an ISO image

Before you install a cluster on infrastructure that you provision, you must create RHCOS machines for it to use. You can use an ISO image to create the machines.

**Prerequisites**

- Obtain the Ignition config files for your cluster.
- Have access to an HTTP server that can be accessed from your computer, and from the machines that you create.
Procedure

1. Upload the control plane, compute, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

   **IMPORTANT**

   If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS images that are required for your preferred method of installing operating system instances from the [RHCOS image mirror](#) page.

   **IMPORTANT**

   The RHCOS images might not change with every release of OpenShift Container Platform. You must download images with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image versions that match your OpenShift Container Platform version if they are available. Use only ISO images for this procedure. RHCOS qcow2 images are not supported for this installation type.

   ISO file names resemble the following example:

   `rhcos-<version>-live.<architecture>.iso`

3. Use the ISO to start the RHCOS installation. Use one of the following installation options:

   - Burn the ISO image to a disk and boot it directly.
   - Use ISO redirection via a LOM interface.

4. Boot the ISO image. You can interrupt the installation boot process to add kernel arguments. However, for this ISO procedure you should use the `coreos-installer` command instead of adding kernel arguments. If you run the live installer without options or interruption, the installer boots up to a shell prompt on the live system, ready for you to install RHCOS to disk.

5. Review the Advanced RHCOS installation reference section for different ways of configuring features, such as networking and disk partitions, before running the `coreos-installer`.

6. Run the `coreos-installer` command. At a minimum, you must identify the Ignition config file location for your node type, and the location of the disk you are installing to. Here is an example:

   ```bash
   $ sudo coreos-installer install \
      --ignition-url=https://host/worker.ign /dev/sda
   ```

7. After RHCOS installs, the system reboots. During the system reboot, it applies the Ignition config file that you specified.

8. Continue to create the other machines for your cluster.
IMPORTANT

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

13.1.11.2. Creating Red Hat Enterprise Linux CoreOS (RHCOS) machines by PXE or iPXE booting

Before you install a cluster that uses manually-provisioned RHCOS nodes, such as bare metal, you must create RHCOS machines for it to use. You can use PXE or iPXE booting to create the machines.

Prerequisites

- Obtain the Ignition config files for your cluster.
- Configure suitable PXE or iPXE infrastructure.
- Have access to an HTTP server that you can access from your computer.

Procedure

1. Upload the master, worker, and bootstrap Ignition config files that the installation program created to your HTTP server. Note the URLs of these files.

IMPORTANT

You can add or change configuration settings in your Ignition configs before saving them to your HTTP server. If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

2. Obtain the RHCOS **kernel**, **initramfs** and **rootfs** files from the RHCOS image mirror page.

IMPORTANT

The RHCOS artifacts might not change with every release of OpenShift Container Platform. You must download artifacts with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Only use the appropriate **kernel**, **initramfs**, and **rootfs** artifacts described below for this procedure. RHCOS qcow2 images are not supported for this installation type.

The file names contain the OpenShift Container Platform version number. They resemble the following examples:

- **kernel**: `rhcos-<version>-live-kernel-<architecture>`
- **initramfs**: `rhcos-<version>-live-initramfs.<architecture>.img`
- **rootfs**: `rhcos-<version>-live-rootfs.<architecture>.img`

3. Upload the additional files that are required for your booting method:

- For traditional PXE, upload the **kernel** and **initramfs** files to your TFTP server and the **rootfs** file to your HTTP server.
For iPXE, upload the kernel, initramfs, and rootfs files to your HTTP server.

**IMPORTANT**

If you plan to add more compute machines to your cluster after you finish installation, do not delete these files.

4. Configure the network boot infrastructure so that the machines boot from their local disks after RHCOS is installed on them.

5. Configure PXE or iPXE installation for the RHCOS images. Modify one of the following example menu entries for your environment and verify that the image and Ignition files are properly accessible:

- **For PXE:**
  ```
  DEFAULT pxeboot
  TIMEOUT 20
  PROMPT 0
  LABEL pxeboot
  KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture>
  APPEND coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
  ```

1. Specify the location of the live kernel file that you uploaded to your HTTP server. The URL must be HTTP, TFTP, or FTP; HTTPS and NFS are not supported.

2. If you use multiple NICs, specify a single interface in the ip option. For example, to use DHCP on a NIC that is named eno1, set `ip=eno1:dhcp`.

3. Specify locations of the RHCOS files that you uploaded to your HTTP server. The initrd parameter value is the location of the initramfs file, the coreos.live.roots_url parameter value is the location of the rootfs file, and the coreos.inst.ignition_url parameter value is the location of the bootstrap Ignition config file. You can also add more kernel arguments to the APPEND line to configure networking or other boot options.

**NOTE**

This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more console= arguments to the APPEND line. For example, add `console=tt00 console=tt0s0` to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?.

- **For iPXE:**
  ```
  kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
  ```
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
<architecture>.img

1 Specify locations of the RHCOS files that you uploaded to your HTTP server. The kernel parameter value is the location of the kernel file, the initrd=main argument is needed for booting on UEFI systems, the coreos.live.rootfs_url parameter value is the location of the rootfs file, and the coreos.inst.ignition_url parameter value is the location of the bootstrap Ignition config file.

2 If you use multiple NICs, specify a single interface in the ip option. For example, to use DHCP on a NIC that is named eno1, set ip=eno1:dhcp.

3 Specify the location of the initramfs file that you uploaded to your HTTP server.

NOTE
This configuration does not enable serial console access on machines with a graphical console. To configure a different console, add one or more console= arguments to the kernel line. For example, add console=tty0 console=ttyS0 to set the first PC serial port as the primary console and the graphical console as a secondary console. For more information, see How does one set up a serial terminal and/or console in Red Hat Enterprise Linux?

6. If you use PXE UEFI, perform the following actions:

   a. Provide the shimx64.efi and grubx64.efi EFI binaries and the grub.cfg file that are required for booting the system.

      • Extract the necessary EFI binaries by mounting the RHCOS ISO to your host and then mounting the images/efiboot.img file to your host:

        $ mkdir -p /mnt/iso

        $ mkdir -p /mnt/efiboot

        $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso

        $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot

        $ umount /mnt/efiboot

      • From the efiboot.img mount point, copy the EFI/redhat/shimx64.efi and EFI/redhat/grubx64.efi files to your TFTP server:

        $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .

        $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .

        $ umount /mnt/efiboot
7. Continue to create the machines for your cluster.

**IMPORTANT**

You must create the bootstrap and control plane machines at this time. If the control plane machines are not made schedulable, which is the default, also create at least two compute machines before you install the cluster.

13.1.11.3. Advanced Red Hat Enterprise Linux CoreOS (RHCOS) installation configuration

A key benefit for manually provisioning the Red Hat Enterprise Linux CoreOS (RHCOS) nodes for OpenShift Container Platform is to be able to do configuration that is not available through default OpenShift Container Platform installation methods. This section describes some of the configurations that you can do using techniques that include:

- Passing kernel arguments to the live installer
- Running **coreos-installer** manually from the live system
13.1.11.3.1. Using advanced networking options for PXE and ISO installations

Networking for OpenShift Container Platform nodes uses DHCP by default to gather all necessary configuration settings. To set up static IP addresses or configure special settings, such as bonding, you can do one of the following:

- Pass special kernel parameters when you boot the live installer.
- Use a machine config to copy networking files to the installed system.
- Configure networking from a live installer shell prompt, then copy those settings to the installed system so that they take effect when the installed system first boots.

To configure a PXE or iPXE installation, use one of the following options:

- See the "Advanced RHCOS installation reference" tables.
- Use a machine config to copy networking files to the installed system.

To configure an ISO installation, use the following procedure.

**Procedure**

1. Boot the ISO installer.

2. From the live system shell prompt, configure networking for the live system using available RHEL tools, such as `nmcli` or `nmtui`.

3. Run the `coreos-installer` command to install the system, adding the `--copy-network` option to copy networking configuration. For example:

   ```
   $ coreos-installer install --copy-network \
   --ignition-url=http://host/worker.ign /dev/sda
   ```

   **IMPORTANT**

   The `--copy-network` option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

4. Reboot into the installed system.

13.1.11.3.2. Disk partitioning

The disk partitions are created on OpenShift Container Platform cluster nodes during the Red Hat Enterprise Linux CoreOS (RHCOS) installation. Each RHCOS node of a particular architecture uses the same partition layout, unless the default partitioning configuration is overridden. During the RHCOS installation, the size of the root file system is increased to use the remaining available space on the target device.
However, there are two cases where you might want to intervene to override the default partitioning when installing an OpenShift Container Platform node:

- Create separate partitions: For greenfield installations on an empty disk, you might want to add separate storage to a partition. This is officially supported for making `/var` or a subdirectory of `/var`, such as `/var/lib/etcd`, a separate partition, but not both.

  **IMPORTANT**

  Kubernetes supports only two filesystem partitions. If you add more than one partition to the original configuration, Kubernetes cannot monitor all of them.

- Retain existing partitions: For a brownfield installation where you are reinstalling OpenShift Container Platform on an existing node and want to retain data partitions installed from your previous operating system, there are both boot arguments and options to `coreos-installer` that allow you to retain existing data partitions.

### 13.1.11.3.2.1. Creating a separate `/var` partition

In general, disk partitioning for OpenShift Container Platform should be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the `/var` partition or a subdirectory of `/var`. For example:

- `/var/lib/containers`: Holds container-related content that can grow as more images and containers are added to a system.

- `/var/lib/etcd`: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.

- `/var`: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a `/var` directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because `/var` must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate `/var` partition by creating a machine config that is inserted during the `openshift-install` preparation phases of an OpenShift Container Platform installation.

#### Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

   ```bash
   $ mkdir $HOME/clusterconfig
   ```

2. Run `openshift-install` to create a set of files in the `manifest` and `openshift` subdirectories. Answer the system questions as you are prompted:

   ```bash
   $ openshift-install create manifests --dir $HOME/clusterconfig
   $ openshift-install create manifests --dir $HOME/clusterconfig
   $ openshift-install create manifests --dir $HOME/clusterconfig
   ```
3. Create a MachineConfig object and add it to a file in the openshift directory. For example, name the file 98-var-partition.yaml, change the disk device name to the name of the storage device on the worker systems, and set the storage size as appropriate. This example places the /var directory on a separate partition:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
        - device: /dev/<device_name>  
          partitions:
            - label: var
              startMiB: <partition_start_offset>  
              sizeMiB: <partition_size>  
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
  systemd:
    units:
      - name: var.mount
        enabled: true
        contents: |
          [Unit]
          Before=local-fs.target
          [Mount]
          What=/dev/disk/by-partlabel/var
          Where=/var
          Options=defaults,prjquota  
          [Install]
          WantedBy=local-fs.target
```

1. The storage device name of the disk that you want to partition.
2. When adding a data partition to the boot disk, a minimum value of 25000 mebibytes is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.
3. The size of the data partition in mebibytes.
The name of the mount unit must match the directory specified in the `Where=` directive. For example, for a filesystem mounted on `/var/lib/containers`, the unit must be named `var-`.

The `prjquota` mount option must be enabled for filesystems used for container storage.

**NOTE**

When creating a separate `/var` partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

4. Run `openshift-install` again to create Ignition configs from a set of files in the `manifest` and `openshift` subdirectories:

```bash
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

Now you can use the Ignition config files as input to the ISO or PXE manual installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

### 13.1.11.3.2.2 Retaining existing partitions

For an ISO installation, you can add options to the `coreos-installer` command line that causes the installer to maintain one or more existing partitions. For a PXE installation, you can APPEND `coreos.inst.*` options to preserve partitions.

Saved partitions might be partitions from an existing OpenShift Container Platform system that has data partitions that you want to keep. Here are a few tips:

- If you save existing partitions, and those partitions do not leave enough space for RHCOS, installation will fail without damaging the saved partitions.
- Identify the disk partitions you want to keep either by partition label or by number.

**For an ISO installation**

This example preserves any partition in which the partition label begins with `data` (`data*`):

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partlabel 'data*' /dev/sda
```

The following example illustrates running the `coreos-installer` in a way that preserves the sixth (6) partition on the disk:

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partindex 6 /dev/sda
```

This example preserves partitions 5 and higher:

```bash
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \ --save-partindex 5- /dev/sda
```
In the previous examples where partition saving is used, `coreos-installer` recreates the partition immediately.

For a PXE installation
This **APPEND** option preserves any partition in which the partition label begins with 'data' ('data*'):

```
coreos.inst.save_partlabel=data*
```

This **APPEND** option preserves partitions 5 and higher:

```
coreos.inst.save_partindex=5-
```

This **APPEND** option preserves partition 6:

```
coreos.inst.save_partindex=6
```

13.1.11.3.3. Identifying Ignition configs

When doing an RHCOS manual installation, there are two types of Ignition configs that you can provide, with different reasons for providing each one:

- **Permanent install Ignition config**: Every manual RHCOS installation needs to pass one of the Ignition config files generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, to carry out the installation.

  **IMPORTANT**
  It is not recommended to modify these files.

  For PXE installations, you pass the Ignition configs on the **APPEND** line using the `coreos.inst.ignition_url=` option. For ISO installations, after the ISO boots to the shell prompt, you identify the Ignition config on the `coreos-installer` command line with the `--ignition-url=` option. In both cases, only HTTP and HTTPS protocols are supported.

- **Live install Ignition config**: This type must be created manually and should be avoided if possible, as it is not supported by Red Hat. With this method, the Ignition config passes to the live install medium, runs immediately upon booting, and performs setup tasks before and/or after the RHCOS system installs to disk. This method should only be used for performing tasks that must be performed once and not applied again later, such as with advanced partitioning that cannot be done using a machine config.

  For PXE or ISO boots, you can create the Ignition config and **APPEND** the `ignition.config.url=` option to identify the location of the Ignition config. You also need to append `ignition.firstboot ignition.platform.id=metal` or the `ignition.config.url` option will be ignored.

13.1.11.3.3.1. Embedding an Ignition config in the RHCOS ISO

You can embed a live install Ignition config directly in an RHCOS ISO image. When the ISO image is booted, the embedded config will be applied automatically.

**Procedure**

1. Download the `coreos-installer` binary from the following image mirror page:
2. Retrieve the RHCOS ISO image and the Ignition config file, and copy them into an accessible directory, such as `/mnt`:

```bash
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. Run the following command to embed the Ignition config into the ISO:

```bash
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \ 
   /mnt/rhcos-<version>-live.x86_64.iso
```

You can now use that ISO to install RHCOS using the specified live install Ignition config.

**IMPORTANT**

Using `coreos-installer iso ignition embed` to embed a file generated by `openshift-installer`, such as `bootstrap.ign`, `master.ign` and `worker.ign`, is unsupported and not recommended.

4. To show the contents of the embedded Ignition config and direct it into a file, run:

```bash
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign

# diff -s bootstrap.ign mybootstrap.ign
```

**Example output**

Files `bootstrap.ign` and `mybootstrap.ign` are identical

5. To remove the Ignition config and return the ISO to its pristine state so you can reuse it, run:

```bash
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

You can now embed another Ignition config into the ISO or use the ISO in its pristine state.

### 13.1.11.3.4. Advanced RHCOS installation reference

This section illustrates the networking configuration and other advanced options that allow you to modify the Red Hat Enterprise Linux CoreOS (RHCOS) manual installation process. The following tables describe the kernel arguments and command-line options you can use with the RHCOS live installer and the `coreos-installer` command.

**Routing and bonding options at RHCOS boot prompt**

If you install RHCOS from an ISO image, you can add kernel arguments manually when you boot that image to configure the node’s networking. If no networking arguments are used, the installation defaults to using DHCP.

**IMPORTANT**

When adding networking arguments, you must also add the `rd.neednet=1` kernel argument.
The following table describes how to use `ip=`, `nameserver=`, and `bond=` kernel arguments for live ISO installs.

**NOTE**

Ordering is important when adding kernel arguments: `ip=`, `nameserver=`, and then `bond=`.

Routing and bonding options for ISO

The following table provides examples for configuring networking of your Red Hat Enterprise Linux CoreOS (RHCOS) nodes. These are networking options that are passed to the `dracut` tool during system boot. For more information about the networking options supported by `dracut`, see the `dracut.cmdline` manual page.

<table>
<thead>
<tr>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To configure an IP address, either use DHCP (<code>ip=dhcp</code>) or set an individual static IP address (<code>ip= </code>&lt;host_ip&gt;``). Then identify the DNS server IP address (<code>nameserver=&lt;dns_ip&gt;</code>) on each node. This example sets:</strong></td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41</code></td>
</tr>
<tr>
<td>● The node's IP address to <strong>10.10.10.2</strong></td>
<td></td>
</tr>
<tr>
<td>● The gateway address to <strong>10.10.10.254</strong></td>
<td></td>
</tr>
<tr>
<td>● The netmask to <strong>255.255.255.0</strong></td>
<td></td>
</tr>
<tr>
<td>● The hostname to <strong>core0.example.com</strong></td>
<td></td>
</tr>
<tr>
<td>● The DNS server address to <strong>4.4.4.41</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Specify multiple network interfaces by specifying multiple <code>ip=</code> entries.</strong></td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code></td>
</tr>
<tr>
<td><strong>Optional: You can configure routes to additional networks by setting an <code>rd.route=</code> value.</strong></td>
<td><code>ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>If the additional network gateway is different from the primary network gateway, the default gateway must be the primary network gateway.</td>
<td>To configure the default gateway:</td>
</tr>
<tr>
<td></td>
<td><code>ip=::10.10.10.254:::</code></td>
</tr>
<tr>
<td>To configure the route for the additional network:</td>
<td><code>rd.route=20.20.20.0/24:20.20.20.254:enp2s0</code></td>
</tr>
<tr>
<td>Disable DHCP on a single interface, such as when there are two or more network interfaces and only one interface is being used.</td>
<td><code>ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none</code></td>
</tr>
<tr>
<td></td>
<td><code>ip=:::core0.example.com:enp2s0:none</code></td>
</tr>
<tr>
<td>Description</td>
<td>Examples</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| You can combine DHCP and static IP configurations on systems with multiple network interfaces. | ip=eth0::dhcp  
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:eth0:none  
To configure a VLAN on a network interface and use a static IP address:  
   ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:eth0.100:none  
   vlan=eth0.100:eth0  
To configure a VLAN on a network interface and to use DHCP:  
   ip=eth0.100:dhcp  
   vlan=eth0.100:eth0  
You can provide multiple DNS servers by adding a nameserver= entry for each server. | nameserver=1.1.1.1  
nameserver=8.8.8.8  
Optional: Bonding multiple network interfaces to a single interface is supported using the bond= option. In these two examples:  
   - The syntax for configuring a bonded interface is:  
     bond=name[:network_interfaces] [:options]  
     name is the bonding device name (bond0), network_interfaces represents a comma-separated list of physical (ethernet) interfaces (em1,em2), and options is a comma-separated list of bonding options. Enter modinfo bonding to see available options.  
     - When you create a bonded interface using bond=, you must specify how the IP address is assigned and other information for the bonded interface.  
   To configure the bonded interface to use DHCP, set the bond’s IP address to dhcp. For example:  
   bond=bond0:em1,em2:mode=active-backup  
   ip=bond0:dhcp  
To configure the bonded interface to use a static IP address, enter the specific IP address you want and related information. For example:  
   bond=bond0:em1,em2:mode=active-backup  
   ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none |
Optional: You can configure VLANs on bonded interfaces by using the `vlan=` parameter.

To configure the bonded interface with a VLAN and to use DHCP:

```plaintext
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

To configure the bonded interface with a VLAN and to use a static IP address:

```plaintext
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

Optional: Network teaming can be used as an alternative to bonding by using the `team=` parameter. In this example:

- The syntax for configuring a team interface is: `team=name[:network_interfaces]`
  - `name` is the team device name (team0) and `network_interfaces` represents a comma-separated list of physical (ethernet) interfaces (em1, em2).

**NOTE**

Teaming is planned to be deprecated when RHCOS switches to an upcoming version of RHEL. For more information, see this Red Hat Knowledgebase Article.

To configure a network team:

```plaintext
team=team0:em1,em2
ip=team0:dhcp
```

coreos.inst boot options for ISO or PXE install

While you can pass most standard installation boot arguments to the live installer, there are several arguments that are specific to the RHCOS live installer.

- For ISO, these options can be added by interrupting the RHCOS installer.
- For PXE or iPXE, these options must be added to the `APPEND` line before starting the PXE kernel. You cannot interrupt a live PXE install.

The following table shows the RHCOS live installer boot options for ISO and PXE installs.

**Table 13.7. coreos.inst boot options**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Argument</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>coreos.inst.install_dev</td>
<td>Required. The block device on the system to install to. It is recommended to use the full path, such as /dev/sda, although sda is allowed.</td>
</tr>
<tr>
<td>coreos.inst.ignition_url</td>
<td>Optional: The URL of the Ignition config to embed into the installed system. If no URL is specified, no Ignition config is embedded.</td>
</tr>
<tr>
<td>coreos.inst.save_partlabel</td>
<td>Optional: Comma-separated labels of partitions to preserve during the install. Glob-style wildcards are permitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.save_partindex</td>
<td>Optional: Comma-separated indexes of partitions to preserve during the install. Ranges m-n are permitted, and either m or n can be omitted. The specified partitions do not need to exist.</td>
</tr>
<tr>
<td>coreos.inst.insecure</td>
<td>Optional: Permits the OS image that is specified by coreos.inst.image_url to be unsigned.</td>
</tr>
<tr>
<td>coreos.inst.image_url</td>
<td>Optional: Download and install the specified RHCOS image.</td>
</tr>
<tr>
<td></td>
<td>- This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
<tr>
<td></td>
<td>- While this argument can be used to install a version of RHCOS that does not match the live media, it is recommended that you instead use the media that matches the version you want to install.</td>
</tr>
<tr>
<td></td>
<td>- If you are using coreos.inst.image_url, you must also use coreos.inst.insecure. This is because the bare-metal media are not GPG-signed for OpenShift Container Platform.</td>
</tr>
<tr>
<td></td>
<td>- Only HTTP and HTTPS protocols are supported.</td>
</tr>
<tr>
<td>coreos.inst.skip_reboot</td>
<td>Optional: The system will not reboot after installing. Once the install finishes, you will receive a prompt that allows you to inspect what is happening during installation. This argument should not be used in production environments and is intended for debugging purposes only.</td>
</tr>
</tbody>
</table>
**coreos-installer options for ISO install**

You can also install RHCOS by invoking the `coreos-installer` command directly from the command line. The kernel arguments in the previous table provide a shortcut for automatically invoking `coreos-installer` at boot time, but you can pass similar arguments directly to `coreos-installer` when running it from a shell prompt.

The following table shows the options and subcommands you can pass to the `coreos-installer` command from a shell prompt during a live install.

**Table 13.8. coreos-installer command-line options, arguments, and subcommands**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-u, --image-url &lt;url&gt;</td>
<td>Specify the image URL manually.</td>
</tr>
<tr>
<td>-f, --image-file &lt;path&gt;</td>
<td>Specify a local image file manually.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>Embed an Ignition config from a file.</td>
</tr>
<tr>
<td>-I, --ignition-url &lt;URL&gt;</td>
<td>Embed an Ignition config from a URL.</td>
</tr>
<tr>
<td>--ignition-hash &lt;digest&gt;</td>
<td>Digest <code>type-value</code> of the Ignition config.</td>
</tr>
<tr>
<td>-p, --platform &lt;name&gt;</td>
<td>Override the Ignition platform ID.</td>
</tr>
<tr>
<td>--append-karg &lt;arg&gt;…</td>
<td>Append the default kernel argument.</td>
</tr>
<tr>
<td>--delete-karg &lt;arg&gt;…</td>
<td>Delete the default kernel argument.</td>
</tr>
</tbody>
</table>
**-n, --copy-network**  
Copy the network configuration from the install environment.

**IMPORTANT**  
The **--copy-network** option only copies networking configuration found under `/etc/NetworkManager/system-connections`. In particular, it does not copy the system hostname.

**--network-dir <path>**  
For use with `-n`. Default is `/etc/NetworkManager/system-connections/`.

**--save-partlabel <lx>..**  
Save partitions with this label glob.

**--save-partindex <id>...**  
Save partitions with this number or range.

**--offline**  
Force offline installation.

**--insecure**  
Skip signature verification.

**--insecure-ignition**  
Allow Ignition URL without HTTPS or hash.

**--architecture <name>**  
Target CPU architecture. Default is `x86_64`.

**--preserve-on-error**  
Do not clear partition table on error.

**-h, --help**  
Print help information.

**Command-line argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;device&gt;</td>
<td>The destination device.</td>
</tr>
</tbody>
</table>

**coreos-installer embedded Ignition commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$ coreos-installer iso ignition embed &lt;options&gt; --ignition-file &lt;file_path&gt; &lt;ISO_image&gt;</code></td>
<td>Embed an Ignition config in an ISO image.</td>
</tr>
<tr>
<td><code>coreos-installer iso ignition show &lt;options&gt; &lt;ISO_image&gt;</code></td>
<td>Show the embedded Ignition config from an ISO image.</td>
</tr>
</tbody>
</table>
coreos-installer iso ignition remove <options> <ISO_image>

Remove the embedded Ignition config from an ISO image.

coreos-installer ISO Ignition options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f, --force</td>
<td>Overwrite an existing Ignition config.</td>
</tr>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

coreos-installer PXE Ignition commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreos-installer pxe ignition wrap &lt;options&gt;</td>
<td>Wrap an Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;image_name&gt;</td>
<td>Show the wrapped Ignition config in an image.</td>
</tr>
<tr>
<td>coreos-installer pxe ignition unwrap &lt;options&gt; &lt;initrd_name&gt;</td>
<td>Show the wrapped Ignition config in an initrd image.</td>
</tr>
</tbody>
</table>

coreos-installer PXE Ignition options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --ignition-file &lt;path&gt;</td>
<td>The Ignition config to be used. Default is stdin.</td>
</tr>
<tr>
<td>-o, --output &lt;path&gt;</td>
<td>Write the ISO to a new output file.</td>
</tr>
<tr>
<td>-h, --help</td>
<td>Print help information.</td>
</tr>
</tbody>
</table>

13.1.12. Creating the cluster

To create the OpenShift Container Platform cluster, you wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the required infrastructure for the cluster.
You obtained the installation program and generated the Ignition config files for your cluster.

- You used the Ignition config files to create RHCOS machines for your cluster.
- Your machines have direct Internet access or have an HTTP or HTTPS proxy available.

Procedure

1. Monitor the bootstrap process:

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \
   --log-level=info
```

   1. For `<installation_directory>`, specify the path to the directory that you stored the installation files in.
   2. To view different installation details, specify `warn`, `debug`, or `error` instead of `info`.

Example output

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources

The command succeeds when the Kubernetes API server signals that it has been bootstrapped on the control plane machines.

2. After bootstrap process is complete, remove the bootstrap machine from the load balancer.

   IMPORTANT

   You must remove the bootstrap machine from the load balancer at this point.
   You can also remove or reformat the machine itself.

13.1.13. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster `kubeconfig` file. The `kubeconfig` file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the `oc` CLI.

Procedure

1. Export the `kubeadmin` credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```
For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

2. Verify you can run `oc` commands successfully using the exported configuration:

   $ oc whoami

   Example output

   ```
   system:admin
   ```

13.1.14. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

   $ oc get nodes

   Example output

   ```
   NAME      STATUS    ROLES   AGE   VERSION
   master-0  Ready     master  63m   v1.19.0
   master-1  Ready     master  63m   v1.19.0
   master-2  Ready     master  64m   v1.19.0
   ```

   The output lists all of the machines that you created.

   **NOTE**

   The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

   $ oc get csr

   Example output

   ```
   NAME        AGE     REQUESTOR                                                                   CONDITION
   csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
   ```
In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

**NOTE**

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. Once the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

**NOTE**

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```bash
  $ oc adm certificate approve <csr_name> 1
  1 <csr_name> is the name of a CSR from the list of current CSRs.
  ```

- To approve all pending CSRs, run the following command:

  ```bash
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

**NOTE**

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:
5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

   - To approve them individually, run the following command for each valid CSR:
     
     ```
     $ oc adm certificate approve <csr_name>  
     ```
     
     `<csr_name>` is the name of a CSR from the list of current CSRs.

   - To approve all pending CSRs, run the following command:
     
     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

   ```
   $ oc get nodes
   ```

   **Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>master-0</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-1</td>
<td>Ready</td>
<td>master</td>
<td>73m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>master-2</td>
<td>Ready</td>
<td>master</td>
<td>74m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-0</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
<tr>
<td>worker-1</td>
<td>Ready</td>
<td>worker</td>
<td>11m</td>
<td>v1.20.0</td>
</tr>
</tbody>
</table>

   **NOTE**

   It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

**Additional information**

- For more information on CSRs, see Certificate Signing Requests.

**13.1.15. Initial Operator configuration**
After the control plane initializes, you must immediately configure some Operators so that they all become available.

Prerequisites

- Your control plane has initialized.

Procedure

1. Watch the cluster components come online:

   $ watch -n5 oc get clusteroperators

Example output

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
<th>SINCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h39m</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h12m</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h15m</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h34m</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h2m</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>6h31m</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>openshift-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h56m</td>
</tr>
<tr>
<td>openshift-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h36m</td>
</tr>
<tr>
<td>openshift-samples</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
<tr>
<td>operator-lifecycle-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-catalog</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>operator-lifecycle-manager-packageserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>3h59m</td>
</tr>
<tr>
<td>service-ca</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>29h</td>
</tr>
<tr>
<td>storage</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>4h30m</td>
</tr>
</tbody>
</table>

2. Configure the Operators that are not available.

13.1.15.1. Disabling the default OperatorHub sources
Operator catalogs that source content provided by Red Hat and community projects are configured for OperatorHub by default during an OpenShift Container Platform installation. In a restricted network environment, you must disable the default catalogs as a cluster administrator.

**Procedure**

- Disable the sources for the default catalogs by adding `disableAllDefaultSources: true` to the `OperatorHub` object:

  ```shell
  $ oc patch OperatorHub cluster --type json \
  -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
  ```

**TIP**

Alternatively, you can use the web console to manage catalog sources. From the Administration → Cluster Settings → Global Configuration → OperatorHub page, click the Sources tab, where you can create, delete, disable, and enable individual sources.

### 13.1.15.2. Image registry removed during installation

On platforms that do not provide shareable object storage, the OpenShift Image Registry Operator boots itself as Removed. This allows `openshift-installer` to complete installations on these platform types.

After installation, you must edit the Image Registry Operator configuration to switch the `managementState` from Removed to Managed.

**NOTE**

The Prometheus console provides an `ImageRegistryRemoved` alert, for example:

"Image Registry has been removed. ImageStreamTags, BuildConfigs and DeploymentConfigs which reference ImageStreamTags may not work as expected. Please configure storage and update the config to Managed state by editing configs.imageregistry.operator.openshift.io."

### 13.1.15.3. Image registry storage configuration

The Image Registry Operator is not initially available for platforms that do not provide default storage. After installation, you must configure your registry to use storage so that the Registry Operator is made available.

Instructions are shown for configuring a persistent volume, which is required for production clusters. Where applicable, instructions are shown for configuring an empty directory as the storage location, which is available for only non-production clusters.

Additional instructions are provided for allowing the image registry to use block storage types by using the `Recreate` rollout strategy during upgrades.

#### 13.1.15.3.1. Configuring registry storage for bare metal and other manual installations

As a cluster administrator, following installation you must configure your registry to use storage.

**Prerequisites**
• Cluster administrator permissions.

• A cluster that uses manually-provisioned Red Hat Enterprise Linux CoreOS (RHCOS) nodes, such as bare metal.

• Persistent storage provisioned for your cluster, such as Red Hat OpenShift Container Storage.

**IMPORTANT**

OpenShift Container Platform supports **ReadWriteOnce** access for image registry storage when you have only one replica. To deploy an image registry that supports high availability with two or more replicas, **ReadWriteMany** access is required.

• Must have 100Gi capacity.

**Procedure**

1. To configure your registry to use storage, change the `spec.storage.pvc` in the `configs.imageregistry/cluster` resource.

   **NOTE**

   When using shared storage, review your security settings to prevent outside access.

2. Verify that you do not have a registry pod:

   ```bash
   $ oc get pod -n openshift-image-registry
   ```

   **NOTE**

   If the storage type is `emptyDIR`, the replica number cannot be greater than **1**.

3. Check the registry configuration:

   ```bash
   $ oc edit configs.imageregistry.operator.openshift.io
   ```

   **Example output**

   ```yaml
   storage:
   pvc:
   claim:
   ```

   Leave the `claim` field blank to allow the automatic creation of an `image-registry-storage` PVC.

4. Check the `clusteroperator` status:

   ```bash
   $ oc get clusteroperator image-registry
   ```

5. Ensure that your registry is set to managed to enable building and pushing of images.
Run:

```bash
$ oc edit configs.imageregistry/cluster
```

Then, change the line

```yaml
managementState: Removed
```

to

```yaml
managementState: Managed
```

13.1.15.3.2. Configuring storage for the image registry in non-production clusters

You must configure storage for the Image Registry Operator. For non-production clusters, you can set the image registry to an empty directory. If you do so, all images are lost if you restart the registry.

**Procedure**

- To set the image registry storage to an empty directory:

```bash
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec": {"storage": {"emptyDir":{}}}}'
```

**WARNING**

Configure this option for only non-production clusters.

If you run this command before the Image Registry Operator initializes its components, the `oc patch` command fails with the following error:

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

Wait a few minutes and run the command again.

13.1.15.3.3. Configuring block registry storage

To allow the image registry to use block storage types during upgrades as a cluster administrator, you can use the **Recreate** rollout strategy.

**IMPORTANT**

Block storage volumes are supported but not recommended for use with the image registry on production clusters. An installation where the registry is configured on block storage is not highly available because the registry cannot have more than one replica.
1. To set the image registry storage as a block storage type, patch the registry so that it uses the `Recreate` rollout strategy and runs with only one (1) replica:

   ```bash
   $ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
       {"rolloutStrategy":"Recreate","replicas":1}}'
   ```

2. Provision the PV for the block storage device, and create a PVC for that volume. The requested block volume uses the ReadWriteOnce (RWO) access mode.

3. Edit the registry configuration so that it references the correct PVC.

### 13.1.16. Completing installation on user-provisioned infrastructure

After you complete the Operator configuration, you can finish installing the cluster on infrastructure that you provide.

**Prerequisites**

- Your control plane has initialized.
- You have completed the initial Operator configuration.

**Procedure**

1. Confirm that all the cluster components are online with the following command:

   ```bash
   $ watch -n5 oc get clusteroperators
   ```

**Example output**

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>AVAILABLE</th>
<th>PROGRESSING</th>
<th>DEGRADED</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINCE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>authentication</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cloud-credential</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>cluster-autoscaler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>config-operator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>console</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>csi-snapshot-controller</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>dns</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>etcd</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>image-registry</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>ingress</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>insights</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-apiserver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>kube-storage-version-migrator</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-api</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-approver</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>machine-config</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>marketplace</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>monitoring</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>network</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>node-tuning</td>
<td>4.6.0</td>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>
Alternatively, the following command notifies you when all of the clusters are available. It also retrieves and displays credentials:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete
```

For `<installation_directory>`, specify the path to the directory that you stored the installation files in.

**Example output**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

The command succeeds when the Cluster Version Operator finishes deploying the OpenShift Container Platform cluster from Kubernetes API server.

**IMPORTANT**

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending `node-bootstrapper` certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for Recovering from expired control plane certificates for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

2. Confirm that the Kubernetes API server is communicating with the pods.

a. To view a list of all pods, use the following command:

```
$ oc get pods --all-namespaces
```

**Example output**

<table>
<thead>
<tr>
<th>NAMESPACE</th>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>openshift-apiserver-operator</td>
<td>openshift-apiserver-operator-85cb746d55-zqhs8</td>
<td>1/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Running</td>
<td>1</td>
<td>9m</td>
</tr>
</tbody>
</table>
b. View the logs for a pod that is listed in the output of the previous command by using the following command:

```
$ oc logs <pod_name> -n <namespace>
```

Specify the pod name and namespace, as shown in the output of the previous command.

If the pod logs display, the Kubernetes API server can communicate with the cluster machines.

### 13.1.17. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.6, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use `subscription watch` to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

### 13.1.18. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting.
- Set up your registry and configure registry storage.
CHAPTER 14. INSTALLATION CONFIGURATION

14.1. SUPPORTED INSTALLATION METHODS FOR DIFFERENT PLATFORMS

You can perform different types of installations on different platforms.

NOTE

Not all installation options are supported for all platforms, as shown in the following tables.

Table 14.1. Installer-provisioned infrastructure options

<table>
<thead>
<tr>
<th></th>
<th>AWS</th>
<th>Azure</th>
<th>GCP</th>
<th>Open Stack</th>
<th>RHV</th>
<th>Bare metal</th>
<th>vSphere</th>
<th>VMC</th>
<th>IBM Z</th>
<th>IBM Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Custom</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network Operator</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Restricted network</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Private clusters</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Existing virtual private networks</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Government regions</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 14.2. CUSTOMIZING NODES

Although directly making changes to OpenShift Container Platform nodes is discouraged, there are times when it is necessary to implement a required low-level security, networking, or performance feature. Direct changes to OpenShift Container Platform nodes can be done by:

- Creating machine configs that are included in manifest files to start up a cluster during `openshift-install`.
- Creating machine configs that are passed to running OpenShift Container Platform nodes via the Machine Config Operator.

The following sections describe features that you might want to configure on your nodes in this way.

#### 14.2.1. Adding day-1 kernel arguments

Although it is often preferable to modify kernel arguments as a day-2 activity, you might want to add kernel arguments to all master or worker nodes during initial cluster installation. Here are some reasons you might want to add kernel arguments during cluster installation so they take effect before the systems first boot up:

- You want to disable a feature, such as SELinux, so it has no impact on the systems when they first come up.
- You need to do some low-level network configuration before the systems start.

---

#### Table 14.2. User-provisioned infrastructure options

<table>
<thead>
<tr>
<th></th>
<th>AWS</th>
<th>Azure</th>
<th>GCP</th>
<th>Open Stack</th>
<th>RHV</th>
<th>Bare metal</th>
<th>vSphere</th>
<th>VMC</th>
<th>IBM Z</th>
<th>IBM Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Network Operator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Restricted netw</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shared VPC hosted outside of cluster project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
To add kernel arguments to master or worker nodes, you can create a `MachineConfig` object and inject that object into the set of manifest files used by Ignition during cluster setup.

For a listing of arguments you can pass to a RHEL 8 kernel at boot time, see Kernel.org kernel parameters. It is best to only add kernel arguments with this procedure if they are needed to complete the initial OpenShift Container Platform installation.

**Procedure**

1. Change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

2. Decide if you want to add kernel arguments to worker or control plane nodes (also known as the master nodes).

3. In the `openshift` directory, create a file (for example, `99-openshift-machineconfig-master-kargs.yaml`) to define a `MachineConfig` object to add the kernel settings. This example adds a `loglevel=7` kernel argument to control plane nodes:

   ```
   $ cat << EOF > 99-openshift-machineconfig-master-kargs.yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
       machineconfiguration.openshift.io/role: master
   name: 99-openshift-machineconfig-master-kargs
   spec:
     kernelArguments:
       - 'loglevel=7'
   EOF
   ``

   You can change `master` to `worker` to add kernel arguments to worker nodes instead. Create a separate YAML file to add to both master and worker nodes.

You can now continue on to create the cluster.

### 14.2.2. Adding kernel modules to nodes

For most common hardware, the Linux kernel includes the device driver modules needed to use that hardware when the computer starts up. For some hardware, however, modules are not available in Linux. Therefore, you must find a way to provide those modules to each host computer. This procedure describes how to do that for nodes in an OpenShift Container Platform cluster.

When a kernel module is first deployed by following these instructions, the module is made available for the current kernel. If a new kernel is installed, the kmods-via-containers software will rebuild and deploy the module so a compatible version of that module is available with the new kernel.

The way that this feature is able to keep the module up to date on each node is by:

- Adding a systemd service to each node that starts at boot time to detect if a new kernel has been installed and
- If a new kernel is detected, the service rebuilds the module and installs it to the kernel
For information on the software needed for this procedure, see the kmods-via-containers github site.

A few important issues to keep in mind:

- This procedure is Technology Preview.
- Software tools and examples are not yet available in official RPM form and can only be obtained for now from unofficial github.com sites noted in the procedure.
- Third-party kernel modules you might add through these procedures are not supported by Red Hat.
- In this procedure, the software needed to build your kernel modules is deployed in a RHEL 8 container. Keep in mind that modules are rebuilt automatically on each node when that node gets a new kernel. For that reason, each node needs access to a yum repository that contains the kernel and related packages needed to rebuild the module. That content is best provided with a valid RHEL subscription.

14.2.2.1. Building and testing the kernel module container

Before deploying kernel modules to your OpenShift Container Platform cluster, you can test the process on a separate RHEL system. Gather the kernel module’s source code, the KVC framework, and the kmod-via-containers software. Then build and test the module. To do that on a RHEL 8 system, do the following:

Procedure

1. Register a RHEL 8 system:
   ```
   # subscription-manager register
   ```

2. Attach a subscription to the RHEL 8 system:
   ```
   # subscription-manager attach --auto
   ```

3. Install software that is required to build the software and container:
   ```
   # yum install podman make git -y
   ```

4. Clone the kmod-via-containers repository:
   a. Create a folder for the repository:
      ```
      $ mkdir kmods; cd kmods
      ```
   b. Clone the repository:
      ```
      $ git clone https://github.com/kmods-via-containers/kmods-via-containers
      ```

5. Install a KVC framework instance on your RHEL 8 build host to test the module. This adds a kmods-via-container systemd service and loads it:
   a. Change to the kmod-via-containers directory:
b. Install the KVC framework instance:

$sudo make install

c. Reload the systemd manager configuration:

$sudo systemctl daemon-reload

6. Get the kernel module source code. The source code might be used to build a third-party module that you do not have control over, but is supplied by others. You will need content similar to the content shown in the kvc-simple-kmod example that can be cloned to your system as follows:

$ cd .. ; git clone https://github.com/kmods-via-containers/kvc-simple-kmod

7. Edit the configuration file, simple-kmod.conf file, in this example, and change the name of the Dockerfile to Dockerfile.rhel:

a. Change to the kvc-simple-kmod directory:

$ cd kvc-simple-kmod

b. Rename the Dockerfile:

$ cat simple-kmod.conf

Example Dockerfile

```
KMOD_CONTAINER_BUILD_CONTEXT="https://github.com/kmods-via-containers/kvc-
simple-kmod.git"
KMOD_CONTAINER_BUILD_FILE=Dockerfile.rhel
KMOD_SOFTWARE_VERSION=dd1a7d4
KMOD_NAMES="simple-kmod simple-procfs-kmod"
```

8. Create an instance of kmods-via-containers@.service for your kernel module, simple-kmod in this example:

$sudo make install

9. Enable the kmods-via-containers@.service instance:

$sudo kmods-via-containers build simple-kmod $(uname -r)

10. Enable and start the systemd service:

$sudo systemctl enable kmods-via-containers@simple-kmod.service --now

a. Review the service status:

$sudo systemctl status kmods-via-containers@simple-kmod.service
Example output

- kmods-via-containers@simple-kmod.service - Kmods Via Containers - simple-kmod
  Loaded: loaded (/etc/systemd/system/kmods-via-containers@.service; enabled; vendor preset: disabled)
  Active: active (exited) since Sun 2020-01-12 23:49:49 EST; 5s ago...

11. To confirm that the kernel modules are loaded, use the \texttt{lsmod} command to list the modules:

\begin{verbatim}
$ lsmod | grep simple_
\end{verbatim}

Example output

\begin{verbatim}
simple_procfs_kmod   16384  0
simple_kmod          16384  0
\end{verbatim}

12. Optional. Use other methods to check that the \texttt{simple-kmod} example is working:

- Look for a "Hello world" message in the kernel ring buffer with \texttt{dmesg}:

\begin{verbatim}
$ dmesg | grep 'Hello world'
\end{verbatim}

Example output

\begin{verbatim}
[ 6420.761332] Hello world from simple_kmod.
\end{verbatim}

- Check the value of \texttt{simple-procfs-kmod} in /proc:

\begin{verbatim}
$ sudo cat /proc/simple-procfs-kmod
\end{verbatim}

Example output

\begin{verbatim}
simple-procfs-kmod number = 0
\end{verbatim}

- Run the \texttt{spkut} command to get more information from the module:

\begin{verbatim}
$ sudo spkut 44
\end{verbatim}

Example output

\begin{verbatim}
KVC: wrapper simple-kmod for 4.18.0-147.3.1.el8_1.x86_64
Running userspace wrapper using the kernel module container...
+ podman run -i --rm --privileged
  simple-kmod-dd1a7d4:4.18.0-147.3.1.el8_1.x86_64 spkut 44
simple-procfs-kmod number = 0
simple-procfs-kmod number = 44
\end{verbatim}

Going forward, when the system boots this service will check if a new kernel is running. If there is a new kernel, the service builds a new version of the kernel module and then loads it. If the module is already built, it will just load it.
14.2.2.2. Provisioning a kernel module to OpenShift Container Platform

Depending on whether or not you must have the kernel module in place when OpenShift Container Platform cluster first boots, you can set up the kernel modules to be deployed in one of two ways:

- **Provision kernel modules at cluster install time (day-1)** You can create the content as a `MachineConfig` object and provide it to `openshift-install` by including it with a set of manifest files.

- **Provision kernel modules via Machine Config Operator (day-2)** If you can wait until the cluster is up and running to add your kernel module, you can deploy the kernel module software via the Machine Config Operator (MCO).

In either case, each node needs to be able to get the kernel packages and related software packages at the time that a new kernel is detected. There are a few ways you can set up each node to be able to obtain that content.

- Provide RHEL entitlements to each node.
- Get RHEL entitlements from an existing RHEL host, from the `/etc/pki/entitlement` directory and copy them to the same location as the other files you provide when you build your Ignition config.
- Inside the Dockerfile, add pointers to a `yum` repository containing the kernel and other packages. This must include new kernel packages as they are needed to match newly installed kernels.

14.2.2.2.1. Provision kernel modules via a `MachineConfig` object

By packaging kernel module software with a `MachineConfig` object, you can deliver that software to worker or master nodes at installation time or via the Machine Config Operator.

First create a base Ignition config that you would like to use. At installation time, the Ignition config will contain the ssh public key to add to the `authorized_keys` file for the `core` user on the cluster. To add the `MachineConfig` object later via the MCO instead, the SSH public key is not required. For both type, the example simple-kmod service creates a systemd unit file, which requires a `kmods-via-containers@simple-kmod.service`.

**NOTE**
The systemd unit is a workaround for an upstream bug and makes sure that the `kmods-via-containers@simple-kmod.service` gets started on boot:

1. Register a RHEL 8 system:
   ```bash
   # subscription-manager register
   ```

2. Attach a subscription to the RHEL 8 system:
   ```bash
   # subscription-manager attach --auto
   ```

3. Install software needed to build the software:
   ```bash
   # yum install podman make git -y
   ```
4. Create an Ignition config file that creates a systemd unit file:
   a. Create a directory to host the Ignition config file:
      
      ```
      $ mkdir kmods; cd kmods
      ```

   b. Create the Ignition config file that creates a systemd unit file:
      
      ```
      $ cat <<EOF > ./baseconfig.ign
      {
      "ignition": { "version": "3.1.0" },
      "passwd": {
      "users": [
      {
      "name": "core",
      "groups": ["sudo"],
      "sshAuthorizedKeys": [
      "ssh-rsa AAAA"
      ]
      }
      ]
      },
      "systemd": {
      "units": [{
      "name": "require-kvc-simple-kmod.service",
      "enabled": true,
      "contents": "[Unit]\nRequires=kmods-via-containers@simple-kmod.service\n\n[Service]\nType=oneshot\nExecStart=/usr/bin/true\n\n[Install]\nWantedBy=multi-user.target"
      }]
      }
      }
      EOF
      ```

   NOTE

   You must add your public SSH key to the `baseconfig.ign` file to use the file during `openshift-install`. The public SSH key is not needed if you create the `MachineConfig` object using the MCO.

5. Create a base MCO YAML snippet that uses the following configuration:

```
$ cat <<EOF > mc-base.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
    name: 10-kvc-simple-kmod
spec:
  config:
EOF
```
NOTE

The `mc-base.yaml` is set to deploy the kernel module on `worker` nodes. To deploy on master nodes, change the role from `worker` to `master`. To do both, you could repeat the whole procedure using different file names for the two types of deployments.

6. Get the `kmods-via-containers` software:
   a. Clone the `kmods-via-containers` repository:
      ```bash
      $ git clone https://github.com/kmods-via-containers/kmods-via-containers
      ```
   b. Clone the `kvc-simple-kmod` repository:
      ```bash
      $ git clone https://github.com/kmods-via-containers/kvc-simple-kmod
      ```

7. Get your module software. In this example, `kvc-simple-kmod` is used:

8. Create a fakeroot directory and populate it with files that you want to deliver via Ignition, using the repositories cloned earlier:
   a. Create the directory:
      ```bash
      $ FAKEROOT=$(mktemp -d)
      ```
   b. Change to the `kmod-via-containers` directory:
      ```bash
      $ cd kmods-via-containers
      ```
   c. Install the KVC framework instance:
      ```bash
      $ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
      ```
   d. Change to the `kvc-simple-kmod` directory:
      ```bash
      $ cd ../kvc-simple-kmod
      ```
   e. Create the instance:
      ```bash
      $ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
      ```

9. Get a tool called `filetranspiler` and dependent software:
   ```bash
   $ cd .. ; sudo yum install -y python3
   git clone https://github.com/ashcrow/filetranspiler.git
   ```

10. Generate a final machine config YAML (`mc.yaml`) and have it include the base Ignition config, base machine config, and the fakeroot directory with files you would like to deliver:
    ```bash
    $ ./filetranspiler/filetranspilor -i ./baseconfig.ign
    -f ${FAKEROOT} --format=yaml --dereference-symlinks
    | sed 's/^/     /' | (cat mc-base.yaml -) > 99-simple-kmod.yaml
    ```
11. If the cluster is not up yet, generate manifest files and add this file to the `openshift` directory. If the cluster is already running, apply the file as follows:

    $ oc create -f 99-simple-kmod.yaml

Your nodes will start the `kmods-via-containers@simple-kmod.service` service and the kernel modules will be loaded.

12. To confirm that the kernel modules are loaded, you can log in to a node (using `oc debug node/<openshift-node>`, then `chroot /host`). To list the modules, use the `lsmod` command:

    $ lsmod | grep simple_

Example output

```
    simple_proofs_kmod     16384  0
    simple_kmod            16384  0
```

### 14.2.3. Encrypting disks during installation

You can enable encryption for the boot disks on the control plane and compute nodes at installation time. OpenShift Container Platform supports the Trusted Platform Module (TPM) v2 and Tang encryption modes.

- **TPM v2**: This is the preferred mode. TPM v2 stores passphrases in a secure cryptoprocessor contained within a server. You can use this mode to prevent the boot disk data on a cluster node from being decrypted if the disk is removed from the server.

- **Tang**: Tang and Clevis are server and client components that enable network-bound disk encryption (NBDE). You can bind the boot disk data on your cluster nodes to a Tang server. This prevents the data from being decrypted unless the nodes are on a secure network where the Tang server can be accessed. Clevis is an automated decryption framework that is used to implement the decryption on the client side.

**IMPORTANT**

The use of Tang encryption mode to encrypt your disks is only supported for bare metal and vSphere installations on user-provisioned infrastructure.

When the TPM v2 or Tang encryption modes are enabled, the RHCOS boot disks are encrypted using the LUKS2 format.

This feature:

- Is available for installer-provisioned infrastructure and user-provisioned infrastructure deployments
- Is supported on Red Hat Enterprise Linux CoreOS (RHCOS) systems only
- Sets up disk encryption during the manifest installation phase so all data written to disk, from first boot forward, is encrypted
- Encrypts data on the root filesystem only (`/dev/mapper/coreos-luks-root` on `/`)
Follow one of the two procedures to enable disk encryption for the nodes in your cluster.

14.2.3.1. Enabling TPM v2 disk encryption

Use the following procedure to enable TPM v2 mode disk encryption during an OpenShift Container Platform installation.

Prerequisites

- You have downloaded the OpenShift Container Platform installation program on your installation node.

Procedure

1. Check to see if TPM v2 encryption needs to be enabled in the BIOS on each node. This is required on most Dell systems. Check the manual for your computer.

2. On your installation node, change to the directory that contains the installation program and generate the Kubernetes manifests for the cluster:

   ```bash
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   Replace `<installation_directory>` with the path to the directory that you want to store the installation files in.

3. Create machine config files to encrypt the boot disks for the control plane or compute nodes using the TPM v2 encryption mode.

   - To configure encryption on the control plane nodes, save the following machine config sample to a file in the `<installation_directory>/openshift` directory. For example, name the file `99-openshift-master-tpmv2-encryption.yaml`:

     ```yaml
     apiVersion: machineconfiguration.openshift.io/v1
     kind: MachineConfig
     metadata:
       name: master-tpm
       labels:
         machineconfiguration.openshift.io/role: master
     spec:
       config:
         ignition:
           version: 3.1.0
         storage:
           files:
             - contents:
                 source: data:text/plain;base64,e30K
                 mode: 420
                 overwrite: true
                 path: /etc/clevis.json
     ```
To configure encryption on the compute nodes, save the following machine config sample to a file in the `<installation_directory>/openshift` directory. For example, name the file `99-openshift-worker-tpmv2-encryption.yaml`:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: worker-tpm
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
      - contents:
        source: data:text/plain;base64,e30K
        mode: 420
        overwrite: true
        path: /etc/clevis.json
```

4. Create a backup copy of the YAML files. The original YAML files are consumed when you create the Ignition config files.

5. Continue with the remainder of the OpenShift Container Platform installation.

14.2.3.2. Enabling Tang disk encryption

Use the following procedure to enable Tang mode disk encryption during an OpenShift Container Platform installation.

Prerequisites

- You have downloaded the OpenShift Container Platform installation program on your installation node.

- You have access to a Red Hat Enterprise Linux (RHEL) 8 machine that can be used to generate a thumbprint of the Tang exchange key.

Procedure

1. Set up a Tang server or access an existing one. See Network-bound disk encryption for instructions.

2. Add kernel arguments to configure networking when you do the Red Hat Enterprise Linux CoreOS (RHCOS) installations for your cluster. For example, to configure DHCP networking, identify `ip=dhcp`, or set static networking when you add parameters to the kernel command line. For both DHCP and static networking, you also must provide the `rd.neednet=1` kernel argument.

   **IMPORTANT**

   Skipping this step causes the second boot to fail.
4. Install the **clevis** package on a RHEL 8 machine, if it is not already installed:

$$\texttt{sudo yum install clevis}$$

5. On the RHEL 8 machine, run the following command to generate a thumbprint of the exchange key. Replace `http://tang.example.com:7500` with the URL of your Tang server:

$$\texttt{clevis-encrypt-tang \{"url":"http://tang.example.com:7500\}\} < /dev/null > /dev/null}$$

   In this example, **tangd.socket** is listening on port **7500** on the Tang server.

   **NOTE**

   The **clevis-encrypt-tang** command is used in this step only to generate a thumbprint of the exchange key. No data is being passed to the command for encryption at this point, so `/dev/null` is provided as an input instead of plain text. The encrypted output is also sent to `/dev/null`, because it is not required for this procedure.

**Example output**

The advertisement contains the following signing keys:

`PLjNyRdGw03zlRoGjQYMahSZGu9`  

The thumbprint of the exchange key.

When the **Do you wish to trust these keys? [ynYN]** prompt displays, type **Y**.

**NOTE**

RHEL 8 provides Clevis version 15, which uses the SHA-1 hash algorithm to generate thumbprints. Some other distributions provide Clevis version 17 or later, which use the SHA-256 hash algorithm for thumbprints. You must use a Clevis version that uses SHA-1 to create the thumbprint, to prevent Clevis binding issues when you install Red Hat Enterprise Linux CoreOS (RHCOS) on your OpenShift Container Platform cluster nodes.

6. Create a Base64 encoded file, replacing the URL of the Tang server (**url**) and thumbprint (**thp**) you just generated:

$$\texttt{\$(cat <<EOM}
\{ 
  "url": "http://tang.example.com:7500",  \\n  "thp": "PLjNyRdGw03zlRoGjQYMahSZGu9" 
\} \) | base64 -w0}$$

   Specify the URL of a Tang server. In this example, **tangd.socket** is listening on port **7500** on the Tang server.
Specify the exchange key thumbprint, which was generated in a preceding step.

Example output

```
ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIiwgCiAidGhwIjogIlBMak55UmRHdzAzemxSb0dqUVINYWhTWkd1OSlgCn0K
```

7. If you have not yet generated the Kubernetes manifests, change to the directory that contains the installation program on your installation node and create them:

Example output

```
$ ./openshift-install create manifests --dir <installation_directory> ①
```

Replace `<installation_directory>` with the path to the directory that you want to store the installation files in.

8. Create machine config files to encrypt the boot disks for the control plane or compute nodes using the Tang encryption mode.

- To configure encryption on the control plane nodes, save the following machine config sample to a file in the `<installation_directory>/openshift` directory. For example, name the file `99-openshift-master-tang-encryption.yaml`:

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: master-tang
labels:
  machineconfiguration.openshift.io/role: master
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
        - contents:
            source: data:text/plain;base64,e30K
            source: data:text/plain;base64,ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIiwgCiAidGhwIjogIlBMak55UmRHdzAzemxSb0dqUVINYWhTWkd1OSlgCn0K
          mode: 420
          overwrite: true
          path: /etc/clevis.json
        kernelArguments:
          - rd.neednet=1 ②
```

1 Specify the Base64 encoded string that was generated in the preceding step.

2 Add the `rd.neednet=1` kernel argument to bring the network up in the initramfs. This argument is required.
To configure encryption on the compute nodes, save the following machine config sample to a file in the `<installation_directory>/openshift` directory. For example, name the file `99-openshift-worker-tang-encryption.yaml`:

```yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: worker-tang
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      files:
        - contents:
            source: data:text/plain;base64,e30K
            source: data:text/plain;base64,ewogInVybCI6ICJodHRwOi8vdGFuZy5leGFtcGxlLmNvbTo3NTAwIiA
gCiAidGhwIjogIlBMak55UmRHdzAzemxSb0dqUVINYWhTWkd1OSIgCn0K
        mode: 420
        overwrite: true
        path: /etc/clevis.json
    kernelArguments:
      - rd.neednet=1

1. Specify the Base64 encoded string that was generated in the preceding step.

2. Add the `rd.neednet=1` kernel argument to bring the network up in the initramfs. This argument is required.

9. Create a backup copy of the YAML files. The original YAML files are consumed when you create the Ignition config files.

10. Continue with the remainder of the OpenShift Container Platform installation.

14.2.4. Configuring a RAID-enabled data volume

You can enable software RAID partitioning to provide an external data volume.

**Procedure**

1. Create a machine config in the `<installation_directory>/openshift` directory that configures a data volume by using software RAID. In this example, it is called `raid1-alt-storage.yaml`:

   ```yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     name: raid1-alt-storage
     labels:
       machineconfiguration.openshift.io/role: worker
   ```
If you choose a different mount point, you must update the unit name to correspond to your mount point. Otherwise the unit will not activate. You can generate a matching unit name with `echo $(systemd-escape -p $mountpoint).mount` where `$mountpoint` is your chosen mount point.

Apply the config to your cluster by running the following command:

```bash
$ oc create -f raid1-alt-storage.yaml
```

14.2.5. Configuring chrony time service
You can set the time server and related settings used by the chrony time service (chronyd) by modifying the contents of the chrony.conf file and passing those contents to your nodes as a machine config.

**Procedure**

1. Create the contents of the chrony.conf file and encode it as base64. For example:

   ```bash
   $ cat << EOF | base64
   pool 0.rhel.pool.ntp.org iburst
   driftfile /var/lib/chrony/drift
   makestep 1.0 3
   rttsync
   logdir /var/log/chrony
   EOF
   
   1 Specify any valid, reachable time source, such as the one provided by your DHCP server. Alternately, you can specify any of the following NTP servers: 1.rhel.pool.ntp.org, 2.rhel.pool.ntp.org, or 3.rhel.pool.ntp.org.

2. Create the MachineConfig object file, replacing the base64 string with the one you just created. This example adds the file to master nodes. You can change it to worker or make an additional MachineConfig for the worker role. Create MachineConfig files for each type of machine that your cluster uses:

   ```bash
   $ cat << EOF > ./99-masters-chrony-configuration.yaml
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfig
   metadata:
     labels:
       machineconfiguration.openshift.io/role: master
   name: 99-masters-chrony-configuration
   spec:
     config:
       ignition:
         config: {}
       security:
         tls: {}
       timeouts: {}
       version: 3.1.0
       networkd: {}
       passwd: {}
     storage:
       files:
       - contents: data:text/plain;charset=utf-8;base64,ICAgIHNlcnZlcjBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGlkL2Nocm9ueS9kcmlmdAogICAgbWFrZXMwICh0cy5wL2Nocm9ueS9kaHJvbnkK
   ```
Specify an octal value mode for the `mode` field in the machine config file. After creating the file and applying the changes, the `mode` is converted to a decimal value. You can check the YAML file with the command `oc get mc <mc-name> -o yaml`.

3. Make a backup copy of the configuration files.

4. Apply the configurations in one of two ways:
   - If the cluster is not up yet, after you generate manifest files, add this file to the `<installation_directory>/openshift` directory, and then continue to create the cluster.
   - If the cluster is already running, apply the file:

     ```bash
     $ oc apply -f ./99-masters-chrony-configuration.yaml
     ```

14.2.6. Additional resources

- See [Support for FIPS cryptography](#) for information on FIPS support.

### 14.3. AVAILABLE CLUSTER CUSTOMIZATIONS

You complete most of the cluster configuration and customization after you deploy your OpenShift Container Platform cluster. A number of configuration resources are available.

You modify the configuration resources to configure the major features of the cluster, such as the image registry, networking configuration, image build behavior, and the identity provider.

For current documentation of the settings that you control by using these resources, use the `oc explain` command, for example `oc explain builds --api-version=config.openshift.io/v1`.

#### 14.3.1. Cluster configuration resources

All cluster configuration resources are globally scoped (not namespaced) and named `cluster`.

<table>
<thead>
<tr>
<th>Resource name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiserver.config.openshift.io</code></td>
<td>Provides API server configuration such as certificates and certificate authorities.</td>
</tr>
<tr>
<td><code>authentication.config.openshift.io</code></td>
<td>Controls the identity provider and authentication configuration for the cluster.</td>
</tr>
</tbody>
</table>
### 14.3.2. Operator configuration resources

These configuration resources are cluster-scoped instances, named `cluster`, which control the behavior of a specific component as owned by a particular Operator.

<table>
<thead>
<tr>
<th>Resource name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>consoles.operator.openshift.io</td>
<td>Controls console appearance such as branding customizations</td>
</tr>
<tr>
<td>config.imageregistry.operator.openshift.io</td>
<td>Configures internal image registry settings such as public routing, log levels, proxy settings, resource constraints, replica counts, and storage type.</td>
</tr>
<tr>
<td>config.samples.operator.openshift.io</td>
<td>Configures the Samples Operator to control which example image streams and templates are installed on the cluster.</td>
</tr>
</tbody>
</table>

### 14.3.3. Additional configuration resources
These configuration resources represent a single instance of a particular component. In some cases, you can request multiple instances by creating multiple instances of the resource. In other cases, the Operator can use only a specific resource instance name in a specific namespace. Reference the component-specific documentation for details on how and when you can create additional resource instances.

<table>
<thead>
<tr>
<th>Resource name</th>
<th>Instance name</th>
<th>Namespace</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alertmanager.monitoring.coreos.com</td>
<td>main</td>
<td>openshift-monitoring</td>
<td>Controls the Alertmanager deployment parameters.</td>
</tr>
<tr>
<td>ingresscontroller.operator.openshift.io</td>
<td>default</td>
<td>openshift-ingress-operator</td>
<td>Configures Ingress Operator behavior such as domain, number of replicas, certificates, and controller placement.</td>
</tr>
</tbody>
</table>

14.3.4. Informational Resources

You use these resources to retrieve information about the cluster. Do not edit these resources directly.

<table>
<thead>
<tr>
<th>Resource name</th>
<th>Instance name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clusterversion.config.openshift.io</td>
<td>version</td>
<td>In OpenShift Container Platform 4.6, you must not customize the ClusterVersion resource for production clusters. Instead, follow the process to update a cluster.</td>
</tr>
<tr>
<td>dns.config.openshift.io</td>
<td>cluster</td>
<td>You cannot modify the DNS settings for your cluster. You can view the DNS Operator status.</td>
</tr>
<tr>
<td>infrastructure.config.openshift.io</td>
<td>cluster</td>
<td>Configuration details allowing the cluster to interact with its cloud provider.</td>
</tr>
<tr>
<td>network.config.openshift.io</td>
<td>cluster</td>
<td>You cannot modify your cluster networking after installation. To customize your network, follow the process to customize networking during installation.</td>
</tr>
</tbody>
</table>

14.3.5. Updating the global cluster pull secret

You can update the global pull secret for your cluster by either replacing the current pull secret or appending a new pull secret.

The procedure is required when users use a separate registry to store images than the registry used during installation.
WARNING
Cluster resources must adjust to the new pull secret, which can temporarily limit the usability of the cluster.

WARNING
Updating the global pull secret will cause node reboots while the Machine Config Operator (MCO) syncs the changes.

Prerequisites
- You have access to the cluster as a user with the `cluster-admin` role.

Procedure
1. Optional: To append a new pull secret to the existing pull secret, complete the following steps:
   a. Enter the following command to download the pull secret:
   ```bash
   $ oc get secret/pull-secret -n openshift-config --template='{{index .data ".dockerconfigjson" | base64decode}}' > <pull_secret_location> 1
   ```
   1. Provide the path to the pull secret file.

   b. Enter the following command to add the new pull secret:
   ```bash
   $ oc registry login --registry="<registry>" --auth-basic="<username>:<password>" --to=<pull_secret_location> 2
   ```
   1. Provide the new registry. You can include multiple repositories within the same registry, for example: 
   ```bash
   --registry="<registry/my-namespace/my-repository>".
   ```
   2. Provide the credentials of the new registry.
   3. Provide the path to the pull secret file.

   Alternatively, you can perform a manual update to the pull secret file.

2. Enter the following command to update the global pull secret for your cluster:
   ```bash
   $ oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=<pull_secret_location> 1
   ```
Provide the path to the new pull secret file.

This update is rolled out to all nodes, which can take some time depending on the size of your cluster. During this time, nodes are drained and pods are rescheduled on the remaining nodes.

14.4. CONFIGURING YOUR FIREWALL

If you use a firewall, you must configure it so that OpenShift Container Platform can access the sites that it requires to function. You must always grant access to some sites, and you grant access to more if you use Red Hat Insights, the Telemetry service, a cloud to host your cluster, and certain build strategies.

14.4.1. Configuring your firewall for OpenShift Container Platform

Before you install OpenShift Container Platform, you must configure your firewall to grant access to the sites that OpenShift Container Platform requires.

There are no special configuration considerations for services running on only controller nodes versus worker nodes.

Procedure

1. Allowlist the following registry URLs:

<table>
<thead>
<tr>
<th>URL</th>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>registry.redhat.io</td>
<td>443, 80</td>
<td>Provides core container images</td>
</tr>
<tr>
<td>quay.io</td>
<td>443, 80</td>
<td>Provides core container images</td>
</tr>
<tr>
<td>*.quay.io</td>
<td>443, 80</td>
<td>Provides core container images</td>
</tr>
<tr>
<td>*.openshiftapps.com</td>
<td>443, 80</td>
<td>Provides Red Hat Enterprise Linux CoreOS (RHCOS) images</td>
</tr>
</tbody>
</table>

When you add a site, such as quay.io, to your allowlist, do not add a wildcard entry, such as *.quay.io, to your denylist. In most cases, image registries use a content delivery network (CDN) to serve images. If a firewall blocks access, then image downloads are denied when the initial download request is redirected to a hostname such as cdn01.quay.io.

CDN hostnames, such as cdn01.quay.io, are covered when you add a wildcard entry, such as *.quay.io, in your allowlist.

2. Allowlist any site that provides resources for a language or framework that your builds require.

3. If you do not disable Telemetry, you must grant access to the following URLs to access Red Hat Insights:

<table>
<thead>
<tr>
<th>URL</th>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>cert-api.access.redhat.com</td>
<td>443, 80</td>
<td>Required for Telemetry</td>
</tr>
</tbody>
</table>
4. If you use Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) to host your cluster, you must grant access to the URLs that provide the cloud provider API and DNS for that cloud:

<table>
<thead>
<tr>
<th>Cloud</th>
<th>URL</th>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS</td>
<td>*.amazonaws.com</td>
<td>443, 80</td>
<td>Required to access AWS services and resources. Review the <a href="https://docs.aws.amazon.com/AmazonS3/latest/userguide/aws-s3-service-endpoints.html">AWS Service Endpoints</a> in the AWS documentation to determine the exact endpoints to allow for the regions that you use.</td>
</tr>
<tr>
<td>GCP</td>
<td>*.googleapis.com</td>
<td>443, 80</td>
<td>Required to access GCP services and resources. Review <a href="https://cloud.google.com/docs/deploying-api">Cloud Endpoints</a> in the GCP documentation to determine the endpoints to allow for your APIs.</td>
</tr>
<tr>
<td></td>
<td>accounts.google.com</td>
<td>443, 80</td>
<td>Required to access your GCP account.</td>
</tr>
<tr>
<td>Azure</td>
<td>management.azure.com</td>
<td>443, 80</td>
<td>Required to access Azure services and resources. Review the <a href="https://docs.microsoft.com/en-us/rest/api/">Azure REST API reference</a> in the Azure documentation to determine the endpoints to allow for your APIs.</td>
</tr>
<tr>
<td></td>
<td>*.blob.core.windows.net</td>
<td>443, 80</td>
<td>Required to download Ignition files.</td>
</tr>
<tr>
<td></td>
<td>login.microsoftonline.com</td>
<td>443, 80</td>
<td>Required to access Azure services and resources. Review the <a href="https://docs.microsoft.com/en-us/rest/api/">Azure REST API reference</a> in the Azure documentation to determine the endpoints to allow for your APIs.</td>
</tr>
</tbody>
</table>

5. Allowlist the following URLs:
<table>
<thead>
<tr>
<th>URL</th>
<th>Port</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>mirror.openshift.com</td>
<td>443, 80</td>
<td>Required to access mirrored installation content and images. This site is also a source of release image signatures, although the Cluster Version Operator needs only a single functioning source.</td>
</tr>
<tr>
<td>storage.googleapis.com/openshift-release</td>
<td>443, 80</td>
<td>A source of release image signatures, although the Cluster Version Operator needs only a single functioning source.</td>
</tr>
<tr>
<td>*.apps.&lt;cluster_name&gt;.&lt;base_domain&gt;</td>
<td>443, 80</td>
<td>Required to access the default cluster routes unless you set an ingress wildcard during installation.</td>
</tr>
<tr>
<td>quayio-production-s3.s3.amazonaws.com</td>
<td>443, 80</td>
<td>Required to access Quay image content in AWS.</td>
</tr>
<tr>
<td>api.openshift.com</td>
<td>443, 80</td>
<td>Required both for your cluster token and to check if updates are available for the cluster.</td>
</tr>
<tr>
<td>art-rhcos-ci.s3.amazonaws.com</td>
<td>443, 80</td>
<td>Required to download Red Hat Enterprise Linux CoreOS (RHCOS) images.</td>
</tr>
<tr>
<td>console.redhat.com/openshift</td>
<td>443, 80</td>
<td>Required for your cluster token.</td>
</tr>
<tr>
<td>registry.access.redhat.com</td>
<td>443, 80</td>
<td>Required for odo CLI.</td>
</tr>
<tr>
<td>sso.redhat.com</td>
<td>443, 80</td>
<td>The <a href="https://console.redhat.com/openshift">https://console.redhat.com/openshift</a> site uses authentication from sso.redhat.com</td>
</tr>
</tbody>
</table>

Operators require route access to perform health checks. Specifically, the authentication and web console Operators connect to two routes to verify that the routes work. If you are the cluster administrator and do not want to allow *.apps.<cluster_name>.<base_domain>, then allow these routes:

- **oauth-openshift.apps.<cluster_name>.<base_domain>**

- **console-openshift-console.apps.<cluster_name>.<base_domain>**, or the hostname that is specified in the `spec.route.hostname` field of the `consoles.operator/cluster` object if the field is not empty.

6. Allowlist the following URLs for optional third-party content:
### 7. NTP Server URLs

If you use a default Red Hat Network Time Protocol (NTP) server, allow the following URLs:

- 1.rhel.pool.ntp.org
- 2.rhel.pool.ntp.org
- 3.rhel.pool.ntp.org

**NOTE**

If you do not use a default Red Hat NTP server, verify the NTP server for your platform and allow it in your firewall.

### 14.5. Configuring a Private Cluster

After you install an OpenShift Container Platform version 4.6 cluster, you can set some of its core components to be private.

#### 14.5.1. About private clusters

By default, OpenShift Container Platform is provisioned using publicly-accessible DNS and endpoints. You can set the DNS, Ingress Controller, and API server to private after you deploy your cluster.

**DNS**

If you install OpenShift Container Platform on installer-provisioned infrastructure, the installation program creates records in a pre-existing public zone and, where possible, creates a private zone for the cluster’s own DNS resolution. In both the public zone and the private zone, the installation program or cluster creates DNS entries for *.apps*, for the Ingress object, and *api*, for the API server.

The *.apps* records in the public and private zone are identical, so when you delete the public zone, the private zone seamlessly provides all DNS resolution for the cluster.

**Ingress Controller**

Because the default Ingress object is created as public, the load balancer is internet-facing and in the public subnets. You can replace the default Ingress Controller with an internal one.

**API server**
CHAPTER 14. INSTALLATION CONFIGURATION

By default, the installation program creates appropriate network load balancers for the API server to
use for both internal and external traffic.
On Amazon Web Services (AWS), separate public and private load balancers are created. The load
balancers are identical except that an additional port is available on the internal one for use within the
cluster. Although the installation program automatically creates or destroys the load balancer based on
API server requirements, the cluster does not manage or maintain them. As long as you preserve the
cluster’s access to the API server, you can manually modify or move the load balancers. For the public
load balancer, port 6443 is open and the health check is configured for HTTPS against the /readyz path.
On Google Cloud Platform, a single load balancer is created to manage both internal and external API
traffic, so you do not need to modify the load balancer.
On Microsoft Azure, both public and private load balancers are created. However, because of limitations
in current implementation, you just retain both load balancers in a private cluster.

14.5.2. Setting DNS to private
After you deploy a cluster, you can modify its DNS to use only a private zone.
Procedure
1. Review the DNS custom resource for your cluster:
$ oc get dnses.config.openshift.io/cluster -o yaml

Example output
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
creationTimestamp: "2019-10-25T18:27:09Z"
generation: 2
name: cluster
resourceVersion: "37966"
selfLink: /apis/config.openshift.io/v1/dnses/cluster
uid: 0e714746-f755-11f9-9cb1-02ff55d8f976
spec:
baseDomain: <base_domain>
privateZone:
tags:
Name: <infrastructure_id>-int
kubernetes.io/cluster/<infrastructure_id>: owned
publicZone:
id: Z2XXXXXXXXXXA4
status: {}
Note that the spec section contains both a private and a public zone.
2. Patch the DNS custom resource to remove the public zone:
$ oc patch dnses.config.openshift.io/cluster --type=merge --patch='{"spec": {"publicZone":
null}}'
dns.config.openshift.io/cluster patched
Because the Ingress Controller consults the DNS definition when it creates Ingress objects,

2225


Because the Ingress Controller consults the DNS definition when it creates Ingress objects, when you create or modify Ingress objects, only private records are created.

**IMPORTANT**

DNS records for the existing Ingress objects are not modified when you remove the public zone.

3. Optional: Review the DNS custom resource for your cluster and confirm that the public zone was removed:

   ```bash
   $ oc get dnses.config.openshift.io/cluster -o yaml
   ```

**Example output**

```yaml
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: "2019-10-25T18:27:09Z"
  generation: 2
  name: cluster
  resourceVersion: "37966"
  selfLink: /apis/config.openshift.io/v1/dnses/cluster
  uid: 0e714746-f755-11f9-9cb1-02f155d8f976
spec:
  baseDomain: <base_domain>
  privateZone:
    tags:
      Name: <infrastructure_id>-int
      kubernetes.io/cluster/<infrastructure_id>-wfpg4: owned
  status: {}
```

**14.5.3. Setting the Ingress Controller to private**

After you deploy a cluster, you can modify its Ingress Controller to use only a private zone.

**Procedure**

1. Modify the default Ingress Controller to use only an internal endpoint:

   ```bash
   $ oc replace --force --wait --filename - <<EOF
   apiVersion: operator.openshift.io/v1
   kind: IngressController
   metadata:
     namespace: openshift-ingress-operator
     name: default
   spec:
     endpointPublishingStrategy:
       type: LoadBalancerService
     loadBalancer:
       scope: Internal
   EOF
   ```
14.5.4. Restricting the API server to private

After you deploy a cluster to Amazon Web Services (AWS) or Microsoft Azure, you can reconfigure the API server to use only the private zone.

Prerequisites

- Install the OpenShift CLI (oc).
- Have access to the web console as a user with admin privileges.

Procedure

1. In the web portal or console for AWS or Azure, take the following actions:
   a. Locate and delete appropriate load balancer component.
      - For AWS, delete the external load balancer. The API DNS entry in the private zone already points to the internal load balancer, which uses an identical configuration, so you do not need to modify the internal load balancer.
      - For Azure, delete the api-internal rule for the load balancer.
   b. Delete the api.$clustername.$yourdomain DNS entry in the public zone.

2. Remove the external load balancers:

   IMPORTANT
   You can run the following steps only for an installer-provisioned infrastructure (IPI) cluster. For a user-provisioned infrastructure (UPI) cluster, you must manually remove or disable the external load balancers.

   a. From your terminal, list the cluster machines:

   Example output

   ingressescontroller.operator.openshift.io "default" deleted
   ingressescontroller.operator.openshift.io/default replaced

   The public DNS entry is removed, and the private zone entry is updated.
You modify the control plane machines, which contain `master` in the name, in the following step.

b. Remove the external load balancer from each control plane machine.

i. Edit a control plane `Machine` object to remove the reference to the external load balancer:

```
$ oc edit machines -n openshift-machine-api <master_name>  
```

Specify the name of the control plane, or `master`, `Machine` object to modify.

ii. Remove the lines that describe the external load balancer, which are marked in the following example, and save and exit the object specification:

```
...  
spec:
  providerSpec:
    value:
      ...
      loadBalancers:
        - name: lk4pj-ext  
          type: network  
        - name: lk4pj-int
          type: network
```

Delete this line.

iii. Repeat this process for each of the machines that contains `master` in the name.
CHAPTER 15. VALIDATING AN INSTALLATION

You can check the status of an OpenShift Container Platform cluster after an installation by following the procedures in this document.

15.1. REVIEWING THE INSTALLATION LOG

You can review a summary of an installation in the OpenShift Container Platform installation log. If an installation succeeds, the information required to access the cluster is included in the log.

Prerequisites

- You have access to the installation host.

Procedure

- Review the `.openshift_install.log` log file in the installation directory on your installation host:

  ```
  $ cat <install_dir>/.openshift_install.log
  ```

Example output

Cluster credentials are included at the end of the log if the installation is successful, as outlined in the following example:

```
... time="2020-12-03T09:50:47Z" level=info msg="Install complete!"
  time="2020-12-03T09:50:47Z" level=info msg="To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'"
  time="2020-12-03T09:50:47Z" level=info msg="Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com"
  time="2020-12-03T09:50:47Z" level=info msg="Login to the console with user: \"kubeadmin\", and password: \"6zYlx-ckbW3-4d2Ne-IWvDF\""
  time="2020-12-03T09:50:47Z" level=debug msg="Time elapsed per stage:"
  time="2020-12-03T09:50:47Z" level=debug msg=" Infrastructure: 6m45s"
  time="2020-12-03T09:50:47Z" level=debug msg=" Bootstrap Complete: 11m30s"
  time="2020-12-03T09:50:47Z" level=debug msg=" Bootstrap Destroy: 1m5s"
  time="2020-12-03T09:50:47Z" level=debug msg=" Cluster Operators: 17m31s"
  time="2020-12-03T09:50:47Z" level=info msg="Time elapsed: 37m26s"
```

15.2. VIEWING THE IMAGE PULL SOURCE

For clusters with unrestricted network connectivity, you can view the source of your pulled images by using a command on a node, such as `crictl images`.

However, for disconnected installations, to view the source of pulled images, you must review the CRI-O logs to locate the **Trying to access** log entry, as shown in the following procedure. Other methods to view the image pull source, such as the `crictl images` command, show the non-mirrored image name, even though the image is pulled from the mirrored location.

Prerequisites

- You have access to the cluster as a user with the `cluster-admin` role.
Procedure

- Review the CRI-O logs for a master or worker node:

  $ oc adm node-logs <node_name> -u crio

Example output

The **Trying to access** log entry indicates where the image is being pulled from.

```
... Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1366]: time="2021-08-05 10:33:21.594930907Z" level=info msg="Pulling image: quay.io/openshift-release-dev/ocp-release:4.8.4-ppc64le" id=abcd713b-d0e1-4844-ac1c-474c5b60c07c name=/runtime.v1alpha2.ImageService/PullImage
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17 02:52:50.194341109Z" level=info msg="Trying to access "li0317gcp1.mirror-registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec98b00628970e974284b6ddaf9a6a086cb9a7a6c31"
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17 02:52:50.226788351Z" level=info msg="Trying to access "li0317gcp1.mirror-registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec98b00628970e974284b6ddaf9a6a086cb9a7a6c31"
...```

The log might show the image pull source twice, as shown in the preceding example.

If your **ImageContentSourcePolicy** object lists multiple mirrors, OpenShift Container Platform attempts to pull the images in the order listed in the configuration, for example:

```
Trying to access "li0317gcp1.mirror-registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec98b00628970e974284b6ddaf9a6a086cb9a7a6c31"
Trying to access "li0317gcp2.mirror-registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec98b00628970e974284b6ddaf9a6a086cb9a7a6c31"
```

15.3. GETTING CLUSTER VERSION, STATUS, AND UPDATE DETAILS

You can view the cluster version and status by running the **oc get clusterversion** command. If the status shows that the installation is still progressing, you can review the status of the Operators for more information.

You can also list the current update channel and review the available cluster updates.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Obtain the cluster version and overall status:
CHAPTER 15. VALIDATING AN INSTALLATION

$ oc get clusterversion

Example output
NAME
VERSION AVAILABLE PROGRESSING SINCE STATUS
version 4.6.4 True
False
6m25s Cluster version is 4.6.4
The example output indicates that the cluster has been installed successfully.
2. If the cluster status indicates that the installation is still progressing, you can obtain more
detailed progress information by checking the status of the Operators:
$ oc get clusteroperators.config.openshift.io
3. View a detailed summary of cluster specifications, update availability, and update history:
$ oc describe clusterversion
4. List the current update channel:
$ oc get clusterversion -o jsonpath='{.items[0].spec}{"\n"}'

Example output
{"channel":"stable-4.6","clusterID":"245539c1-72a3-41aa-9cec72ed8cf25c5c","upstream":"https://api.openshift.com/api/upgrades_info/v1/graph"}
5. Review the available cluster updates:
$ oc adm upgrade

Example output
Cluster version is 4.6.4
Updates:
VERSION IMAGE
4.6.6 quay.io/openshift-release-dev/ocprelease@sha256:c7e8f18e8116356701bd23ae3a23fb9892dd5ea66c8300662ef30563d7104f3
9
Additional resources
See Querying Operator status after installation for more information about querying Operator
status if your installation is still progressing.
See Troubleshooting Operator issues for information about investigating issues with Operators.
See Updating a cluster for more information on updating your cluster.

See OpenShift Container Platform upgrade channels and releases for an overview about

2231


15.4. QUERYING THE STATUS OF THE CLUSTER NODES BY USING THE CLI

You can verify the status of the cluster nodes after an installation.

Prerequisites

- You have access to the cluster as a user with the `cluster-admin` role.
- You have installed the OpenShift CLI (`oc`).

Procedure

1. List the status of the cluster nodes. Verify that the output lists all of the expected control plane and compute nodes and that each node has a **Ready** status:

   ```
   $ oc get nodes
   
   NAME                          STATUS   ROLES    AGE   VERSION
   compute-1.example.com         Ready    worker   33m   v1.19.0+9f84db3
   control-plane-1.example.com   Ready    master   41m   v1.19.0+9f84db3
   control-plane-2.example.com   Ready    master   45m   v1.19.0+9f84db3
   compute-2.example.com         Ready    worker   38m   v1.19.0+9f84db3
   compute-3.example.com         Ready    worker   33m   v1.19.0+9f84db3
   control-plane-3.example.com   Ready    master   41m   v1.19.0+9f84db3
   
   Example output
   
   NAME                          STATUS ROLES AGE VERSION
   compute-1.example.com         Ready  worker  33m  v1.19.0+9f84db3
   control-plane-1.example.com   Ready  master  41m  v1.19.0+9f84db3
   control-plane-2.example.com   Ready  master  45m  v1.19.0+9f84db3
   compute-2.example.com         Ready  worker  38m  v1.19.0+9f84db3
   compute-3.example.com         Ready  worker  33m  v1.19.0+9f84db3
   control-plane-3.example.com   Ready  master  41m  v1.19.0+9f84db3
   ```

2. Review CPU and memory resource availability for each cluster node:

   ```
   $ oc adm top nodes
   
   NAME                          CPU(cores) CPU%  MEMORY(bytes) MEMORY%
   compute-1.example.com         128m   8%    1132Mi    16%
   control-plane-1.example.com   801m   22%   3471Mi    23%
   control-plane-2.example.com   1718m  49%   6085Mi    40%
   compute-2.example.com         935m   62%   5178Mi    75%
   compute-3.example.com         111m   7%    1131Mi    16%
   control-plane-3.example.com   942m   26%   4100Mi    27%
   
   Example output
   
   NAME                          CPU(cores) CPU%  MEMORY(bytes) MEMORY%
   compute-1.example.com         128m   8%    1132Mi    16%
   control-plane-1.example.com   801m   22%   3471Mi    23%
   control-plane-2.example.com   1718m  49%   6085Mi    40%
   compute-2.example.com         935m   62%   5178Mi    75%
   compute-3.example.com         111m   7%    1131Mi    16%
   control-plane-3.example.com   942m   26%   4100Mi    27%
   ```

Additional resources

- See [Verifying node health](#) for more details about reviewing node health and investigating node issues.
15.5. REVIEWING THE CLUSTER STATUS FROM THE OPENSSHIFT CONTAINER PLATFORM WEB CONSOLE

You can review the following information in the **Overview** page in the OpenShift Container Platform web console:

- The general status of your cluster
- The status of the control plane, cluster Operators, and storage
- CPU, memory, file system, network transfer, and pod availability
- The API address of the cluster, the cluster ID, and the name of the provider
- Cluster version information
- Cluster update status, including details of the current update channel and available updates
- A cluster inventory detailing node, pod, storage class, and persistent volume claim (PVC) information
- A list of ongoing cluster activities and recent events

**Prerequisites**

- You have access to the cluster as a user with the **cluster-admin** role.

**Procedure**

- In the **Administrator** perspective, navigate to **Home → Overview**.

15.6. REVIEWING THE CLUSTER STATUS FROM RED HAT OPENSIFT CLUSTER MANAGER

You can review detailed information about the status of your cluster in the OpenShift Cluster Manager.

**Prerequisites**

- You have access to the cluster as a user with the **cluster-admin** role.

**Procedure**

1. In the **Administrator** perspective, navigate to **Home → Overview → Details → OpenShift Cluster Manager** to open the **Overview** page for the cluster in **OpenShift Cluster Manager**.

   **NOTE**

   Alternatively, you can navigate to **OpenShift Cluster Manager** directly and select your cluster ID from the list of available clusters.

2. In the **Overview** page, review the following information about your cluster:

   - vCPU and memory availability and resource usage
• The cluster ID, status, type, location, and the provider name
• Node counts by node type
• Cluster version details, the creation date of the cluster, and the name of the cluster owner
• The life cycle support status of the cluster
• Subscription information, including the service level agreement (SLA) status, the subscription unit type, the production status of the cluster, the subscription obligation, and the service level
• A cluster history

3. Navigate to the Monitoring page to review the following information:
• A list of any issues that have been detected
• A list of alerts that are firing
• The cluster Operator status and version
• Cluster resource usage

4. Navigate to the Insights page to review the following information provided by Red Hat Insights:
• Potential issues that your cluster might be exposed to, categorized by risk level
• Health-check status by category

Additional resources

• See Using Insights to identify issues with your cluster for more information about reviewing potential issues with your cluster.

15.7. CHECKING CLUSTER RESOURCE AVAILABILITY AND UTILIZATION

OpenShift Container Platform provides a comprehensive set of monitoring dashboards that help you understand the state of cluster components.

In the Administrator perspective, you can access dashboards for core OpenShift Container Platform components, including:

• etcd
• Kubernetes compute resources
• Kubernetes network resources
• Prometheus
• Dashboards relating to cluster and node performance
Prerequisites

- You have access to the cluster as a user with the `cluster-admin` role.

Procedure

1. In the Administrator perspective in the OpenShift Container Platform web console, navigate to Monitoring → Dashboards.

2. Choose a dashboard in the Dashboard list. Some dashboards, such as the `etcd` dashboard, produce additional sub-menus when selected.

3. Optional: Select a time range for the graphs in the Time Range list.

4. Optional: Select a Refresh Interval

5. Hover over each of the graphs within a dashboard to display detailed information about specific items.

Additional resources

- See Monitoring overview for more information about the OpenShift Container Platform monitoring stack.

15.8. LISTING ALERTS THAT ARE FIRING

Alerts provide notifications when a set of defined conditions are true in an OpenShift Container Platform cluster. You can review the alerts that are firing in your cluster by using the Alerting UI in the OpenShift Container Platform web console.
Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. In the **Administrator** perspective, navigate to the **Monitoring → Alerting → Alerts** page.

2. Review the alerts that are firing, including their **Severity**, **State**, and **Source**.

3. Select an alert to view more detailed information in the **Alert Details** page.

Additional resources

- See **Managing alerts** for further details about alerting in OpenShift Container Platform.

15.9. NEXT STEPS

- See **Troubleshooting installations** if you experience issues when installing your cluster.

- After installing OpenShift Container Platform, you can **further expand and customize your cluster**.
CHAPTER 16. TROUBLESHOOTING INSTALLATION ISSUES

To assist in troubleshooting a failed OpenShift Container Platform installation, you can gather logs from the bootstrap and control plane, or master, machines. You can also get debug information from the installation program.

16.1. PREREQUISITES

- You attempted to install an OpenShift Container Platform cluster, and installation failed.

16.2. GATHERING LOGS FROM A FAILED INSTALLATION

If you gave an SSH key to your installation program, you can gather data about your failed installation.

**NOTE**

You use a different command to gather logs about an unsuccessful installation than to gather logs from a running cluster. If you must gather logs from a running cluster, use the `oc adm must-gather` command.

Prerequisites

- Your OpenShift Container Platform installation failed before the bootstrap process finished. The bootstrap node is running and accessible through SSH.

- The `ssh-agent` process is active on your computer, and you provided the same SSH key to both the `ssh-agent` process and the installation program.

- If you tried to install a cluster on infrastructure that you provisioned, you must have the fully qualified domain names of the bootstrap and control plane nodes (also known as the master nodes).

Procedure

1. Generate the commands that are required to obtain the installation logs from the bootstrap and control plane machines:

   - If you used installer-provisioned infrastructure, change to the directory that contains the installation program and run the following command:

     ```bash
     $ ./openshift-install gather bootstrap --dir <installation_directory> 1
     
     installation_directory is the directory you specified when you ran `./openshift-install create cluster`. This directory contains the OpenShift Container Platform definition files that the installation program creates.
     
     For installer-provisioned infrastructure, the installation program stores information about the cluster, so you do not specify the hostnames or IP addresses.
     
   - If you used infrastructure that you provisioned yourself, change to the directory that contains the installation program and run the following command:

     ```bash
     $ ./openshift-install gather bootstrap --dir <installation_directory> \\ 1
     ```
For `installation_directory`, specify the same directory you specified when you ran 
`./openshift-install create cluster`. This directory contains the OpenShift Container 
Platform definition files that the installation program creates.

`<bootstrap_address>` is the fully qualified domain name or IP address of the cluster’s 
bootstrap machine.

For each control plane, or master, machine in your cluster, replace 
`<master_*_address>` with its fully qualified domain name or IP address.

NOTE

A default cluster contains three control plane machines. List all of your 
control plane machines as shown, no matter how many your cluster uses.

Example output

```
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "<installation_directory>/log-bundle-
<timestamp>.tar.gz"
```

If you open a Red Hat support case about your installation failure, include the compressed logs in the case.

16.3. MANUALLY GATHERING LOGS WITH SSH ACCESS TO YOUR 
HOST(S)

Manually gather logs in situations where `must-gather` or automated collection methods do not work.

Prerequisites

- You must have SSH access to your host(s).

Procedure

1. Collect the `bootkube.service` service logs from the bootstrap host using the `journalctl` 
command by running:

```
$ journalctl -b -f -u bootkube.service
```

2. Collect the bootstrap host’s container logs using the podman logs. This is shown as a loop to get 
all of the container logs from the host:

```
$ for pod in $(sudo podman ps -a -q); do sudo podman logs $pod; done
```

3. Alternatively, collect the host’s container logs using the `tail` command by running:
4. Collect the `kubelet.service` and `crio.service` service logs from the master and worker hosts using the `journalctl` command by running:

```bash
$ journalctl -b -f -u kubelet.service -u crio.service
```

5. Collect the master and worker host container logs using the `tail` command by running:

```bash
$ sudo tail -f /var/log/containers/*
```

### 16.4. MANUALLY GATHERING LOGS WITHOUT SSH ACCESS TO YOUR HOST(S)

Manually gather logs in situations where `must-gather` or automated collection methods do not work.

If you do not have SSH access to your node, you can access the systems journal to investigate what is happening on your host.

**Prerequisites**

- Your OpenShift Container Platform installation must be complete.
- Your API service is still functional.
- You have system administrator privileges.

**Procedure**

1. Access `journald` unit logs under `/var/log` by running:

```bash
$ oc adm node-logs --role=master -u kubelet
```

2. Access host file paths under `/var/log` by running:

```bash
$ oc adm node-logs --role=master --path=openshift-apiserver
```

### 16.5. GETTING DEBUG INFORMATION FROM THE INSTALLATION PROGRAM

You can use any of the following actions to get debug information from the installation program.

- Look at debug messages from a past installation in the hidden `.openshift_install.log` file. For example, enter:

```bash
$ cat ~/<installation_directory>/.openshift_install.log
```

For `installation_directory`, specify the same directory you specified when you ran 
`./openshift-install create cluster`.  

---

1. For `installation_directory`, specify the same directory you specified when you ran 
`./openshift-install create cluster`.  

---

---

---

---
- Change to the directory that contains the installation program and re-run it with `--log-level=debug`:

```bash
$ ./openshift-install create cluster --dir <installation_directory> --log-level debug
```

For `installation_directory`, specify the same directory you specified when you ran `./openshift-install create cluster`.

### 16.6. REINSTALLING THE OPENSHIFT CONTAINER PLATFORM CLUSTER

If you are unable to debug and resolve issues in the failed OpenShift Container Platform installation, consider installing a new OpenShift Container Platform cluster. Before starting the installation process again, you must complete thorough cleanup. For a user-provisioned infrastructure (UPI) installation, you must manually destroy the cluster and delete all associated resources. The following procedure is for an installer-provisioned infrastructure (IPI) installation.

**Procedure**

1. Destroy the cluster and remove all the resources associated with the cluster, including the hidden installer state files in the installation directory:

```bash
$ ./openshift-install destroy cluster --dir <installation_directory>
```

`<installation_directory>` is the directory you specified when you ran `./openshift-install create cluster`. This directory contains the OpenShift Container Platform definition files that the installation program creates.

2. Before reinstalling the cluster, delete the installation directory:

```bash
$ rm -rf <installation_directory>
```

3. Follow the procedure for installing a new OpenShift Container Platform cluster.
CHAPTER 17. SUPPORT FOR FIPS CRYPTOGRAPHY

You can install an OpenShift Container Platform cluster that uses FIPS Validated / Modules in Process cryptographic libraries on the x86_64 architecture.

For the Red Hat Enterprise Linux CoreOS (RHCOS) machines in your cluster, this change is applied when the machines are deployed based on the status of an option in the install-config.yaml file, which governs the cluster options that a user can change during cluster deployment. With Red Hat Enterprise Linux (RHEL) machines, you must enable FIPS mode when you install the operating system on the machines that you plan to use as worker machines. These configuration methods ensure that your cluster meet the requirements of a FIPS compliance audit: only FIPS Validated / Modules in Process cryptography packages are enabled before the initial system boot.

Because FIPS must be enabled before the operating system that your cluster uses boots for the first time, you cannot enable FIPS after you deploy a cluster.

17.1. FIPS VALIDATION IN OPENSSHIFT CONTAINER PLATFORM

OpenShift Container Platform uses certain FIPS Validated / Modules in Process modules within RHEL and RHCOS for the operating system components that it uses. See RHEL7 core crypto components. For example, when users SSH into OpenShift Container Platform clusters and containers, those connections are properly encrypted.

OpenShift Container Platform components are written in Go and built with Red Hat’s golang compiler. When you enable FIPS mode for your cluster, all OpenShift Container Platform components that require cryptographic signing call RHEL and RHCOS cryptographic libraries.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIPS support in RHEL 7 operating systems.</td>
<td>The FIPS implementation does not offer a single function that both computes hash functions and validates the keys that are based on that hash. This limitation will continue to be evaluated and improved in future OpenShift Container Platform releases.</td>
</tr>
<tr>
<td>FIPS support in CRI-O runtimes.</td>
<td></td>
</tr>
<tr>
<td>FIPS support in OpenShift Container Platform services.</td>
<td></td>
</tr>
<tr>
<td>FIPS Validated / Modules in Process cryptographic module and algorithms that are obtained from RHEL 7 and RHCOS binaries and images.</td>
<td></td>
</tr>
<tr>
<td>Use of FIPS compatible golang compiler.</td>
<td>TLS FIPS support is not complete but is planned for future OpenShift Container Platform releases.</td>
</tr>
<tr>
<td>FIPS support across multiple architectures.</td>
<td>FIPS is currently only supported on OpenShift Container Platform deployments using the x86_64 architecture.</td>
</tr>
</tbody>
</table>

17.2. FIPS SUPPORT IN COMPONENTS THAT THE CLUSTER USES
Although the OpenShift Container Platform cluster itself uses FIPS Validated / Modules in Process modules, ensure that the systems that support your OpenShift Container Platform cluster use FIPS Validated / Modules in Process modules for cryptography.

### 17.2.1. etcd

To ensure that the secrets that are stored in etcd use FIPS Validated / Modules in Process encryption, boot the node in FIPS mode. After you install the cluster in FIPS mode, you can encrypt the etcd data by using the FIPS-approved aes cbc cryptographic algorithm.

### 17.2.2. Storage

For local storage, use RHEL-provided disk encryption or Container Native Storage that uses RHEL-provided disk encryption. By storing all data in volumes that use RHEL-provided disk encryption and enabling FIPS mode for your cluster, both data at rest and data in motion, or network data, are protected by FIPS Validated / Modules in Process encryption. You can configure your cluster to encrypt the root filesystem of each node, as described in Customizing nodes.

### 17.2.3. Runtimes

To ensure that containers know that they are running on a host that is using FIPS Validated / Modules in Process cryptography modules, use CRI-O to manage your runtimes. CRI-O supports FIPS mode, in that it configures the containers to know that they are running in FIPS mode.

### 17.3. INSTALLING A CLUSTER IN FIPS MODE

To install a cluster in FIPS mode, follow the instructions to install a customized cluster on your preferred infrastructure. Ensure that you set fips: true in the install-config.yaml file before you deploy your cluster.

- Amazon Web Services
- Microsoft Azure
- Bare metal
- Google Cloud Platform
- Red Hat OpenStack Platform (RHOSP)
- VMware vSphere

**NOTE**

If you are using Azure File storage, you cannot enable FIPS mode.

To apply AES CBC encryption to your etcd data store, follow the Encrypting etcd data process after you install your cluster.

If you add RHEL nodes to your cluster, ensure that you enable FIPS mode on the machines before their initial boot. See Adding RHEL compute machines to an OpenShift Container Platform cluster and Enabling FIPS Mode in the RHEL 7 documentation.