OpenShift Container Platform 4.2

Installing on GCP

Installing OpenShift Container Platform 4.2 GCP clusters

Last Updated: 2020-05-15

OpenShift Container Platform 4.2 Installing on GCP

Installing OpenShift Container Platform 4.2 GCP clusters

Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document provides instructions for installing and uninstalling OpenShift Container Platform 4.2 clusters on Google Cloud Platform.

Table of Contents

CHAPTER 1. INSTALLING ON GCP 4
1.1. CONFIGURING A GCP PROJECT 4
1.1.1. Creating a GCP project 4
1.1.2. Enabling API services in GCP 4
1.1.3. Configuring DNS for GCP 5
1.1.4. GCP account limits 5
1.1.5. Creating a service account in GCP 7
1.1.5.1. Required GCP permissions 8
1.1.6. Supported GCP regions 9
1.2. INSTALLING A CLUSTER QUICKLY ON GCP 9
1.2.1. Internet and Telemetry access for OpenShift Container Platform 10
1.2.2. Generating an SSH private key and adding it to the agent 10
1.2.3. Obtaining the installation program 11
1.2.4. Deploy the cluster 12
1.2.5. Installing the CLI 14
1.2.6. Logging in to the cluster 14
1.3. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS 15
1.3.1. Internet and Telemetry access for OpenShift Container Platform 15
1.3.2. Generating an SSH private key and adding it to the agent 16
1.3.3. Obtaining the installation program 17
1.3.4. Creating the installation configuration file 18
1.3.4.1. Installation configuration parameters 19
1.3.4.2. Sample customized install-config.yaml file for GCP 22
1.3.5. Deploy the cluster 23
1.3.6. Installing the CLI 24
1.3.7. Logging in to the cluster 25
1.4. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS 26
1.4.1. Internet and Telemetry access for OpenShift Container Platform 26
1.4.2. Generating an SSH private key and adding it to the agent 27
1.4.3. Obtaining the installation program 28
1.4.4. Creating the installation configuration file 28
1.4.4.1. Installation configuration parameters 30
1.4.4.2. Network configuration parameters 33
1.4.4.3. Sample customized install-config.yaml file for GCP 33
1.4.5. Modifying advanced network configuration parameters 35
1.4.6. Cluster Network Operator custom resource (CR) 37
1.4.6.1. Configuration parameters for OpenShift SDN 37
1.4.6.2. Cluster Network Operator example CR 38
1.4.7. Deploy the cluster 39
1.4.8. Installing the CLI 39
1.4.9. Logging in to the cluster 40
1.5. INSTALLING A CLUSTER ON GCP USING DEPLOYMENT MANAGER TEMPLATES 41
1.5.1. Certificate signing requests management 41
1.5.2. Configuring your GCP project 41
1.5.2.1. Creating a GCP project 41
1.5.2.2. Enabling API services in GCP 41
1.5.2.3. Configuring DNS for GCP 42
1.5.2.4. GCP account limits 43
1.5.2.5. Creating a service account in GCP 44
1.5.2.5.1. Required GCP permissions 45
1.5.2.6. Supported GCP regions 46

1.5.2.7. Installing and configuring CLI tools for GCP 47
1.5.3. Creating the installation files for GCP 47
1.5.3.1. Creating the installation configuration file 47
1.5.3.2. Configuring the cluster-wide proxy during installation 49
1.5.3.3. Creating the Kubernetes manifest and Ignition config files 50
1.5.4. Exporting common variables 52
1.5.4.1. Extracting the infrastructure name 52
1.5.4.2. Exporting common variables for Deployment Manager templates 53
1.5.5. Creating a VPC in GCP 54
1.5.5.1. Deployment Manager template for the VPC 55
1.5.6. Creating networking and load balancing components in GCP 56
1.5.6.1. Deployment Manager template for the network and load balancers 58
1.5.7. Creating firewall rules and IAM roles in GCP 59
1.5.7.1. Deployment Manager template for firewall rules and IAM roles 61
1.5.8. Creating the RHCOS cluster image for the GCP infrastructure 64
1.5.9. Creating the bootstrap machine in GCP 64
1.5.9.1. Deployment Manager template for the bootstrap machine 66
1.5.10. Creating the control plane machines in GCP 67
1.5.10.1. Deployment Manager template for control plane machines 70
1.5.11. Wait for bootstrap completion and remove bootstrap resources in GCP 72
1.5.12. Creating additional worker machines in GCP 73
1.5.12.1. Deployment Manager template for worker machines 74
1.5.13. Installing the CLI 75
1.5.14. Logging in to the cluster 76
1.5.15. Approving the CSRs for your machines 76
1.5.16. Optional: Adding the ingress DNS records 78
1.5.17. Completing a GCP installation on user-provisioned infrastructure 79
1.6. UNINSTALLING A CLUSTER ON GCP 81
1.6.1. Removing a cluster that uses installer-provisioned infrastructure 81

CHAPTER 1. INSTALLING ON GCP

1.1. CONFIGURING A GCP PROJECT

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

1.1.1. 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.

1.1.2. 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 1.1. Required API services

API service	Console service name
Compute Engine API	**compute.googleapis.com**
Google Cloud APIs	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management (IAM) API	**iam.googleapis.com**
Service Management API	**servicemanagement.googleapis.com**

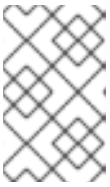
API service	Console service name
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

1.1.3. 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.

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.

1.1.4. 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 1.2. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	5	0
Firewall Rules	Compute	Global	11	1
Forwarding Rules	Compute	Global	2	0
In-use global IP addresses	Compute	Global	4	1
Health checks	Compute	Global	3	0
Images	Compute	Global	1	0
Networks	Compute	Global	2	0
Static IP addresses	Compute	Region	4	1
Routers	Compute	Global	1	0
Routes	Compute	Global	2	0
Subnetworks	Compute	Global	2	0
Target Pools	Compute	Global	3	0
CPUs	Compute	Region	28	4
Persistent Disk SSD (GB)	Compute	Region	896	128

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.

1.1.5. 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 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.

1.1.5.1. 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:

Required roles for the installation program

- Compute Admin
- DNS Administrator
- 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 1.3. GCP service account permissions

Account	Roles
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

1.1.6. 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-south1 (Mumbai, India)
- asia-southeast1 (Jurong West, Singapore)
- 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)

Next steps

- Install an OpenShift Container Platform cluster on GCP. You can install a customized cluster or quickly install a cluster with default options.

1.2. INSTALLING A CLUSTER QUICKLY ON GCP

In OpenShift Container Platform version 4.2, you can install a cluster on Google Cloud Platform (GCP) that uses the default configuration options.

Prerequisites

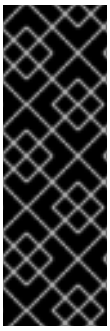
- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

1.2.1. Internet and Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.2, you require access to the internet to install and entitle your cluster. The Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, also requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to the Red Hat OpenShift Cluster Manager. From there, you can allocate entitlements to your cluster.

You must have internet access to:

- Access the Red Hat OpenShift Cluster Manager page to download the installation program and perform subscription management and entitlement. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster. If the Telemetry service cannot entitle your cluster, you must manually entitle it on the Cluster registration page.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

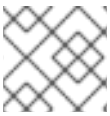


IMPORTANT

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.

1.2.2. 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 to the installation program.



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 rsa -b 4096 -N '' \
    -f <path>/<file_name> (1)
```

(1) Specify the path and file name, such as ~/**.ssh/id_rsa**, of the SSH key.

   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:

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

Agent pid 31874
```

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

```
$ ssh-add <path>/<file_name> (1)

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.

1.2.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on

a local computer.

Prerequisites

- You must install the cluster from a computer that uses Linux or macOS.
- You need 500 MB of local disk space to download the installation program.

Procedure

1. Access the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. 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 both the installation program and the files that the installation program creates after you finish installing the cluster.

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

```
$ tar xvf <installation_program>.tar.gz
```

4. From the Pull Secret page on the Red Hat OpenShift Cluster Manager site, download your installation pull secret as a **.txt** file. 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.

1.2.4. Deploy 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

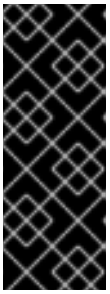
- Configure an account with the cloud platform that hosts your cluster.
- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

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. Run the installation program:

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

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 on, 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.

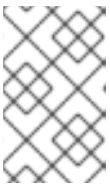
d. Select the project ID to provision the cluster in. The default value is specified by the service account that you configured.

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 that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site.



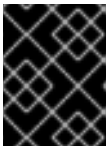
NOTE

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.

IMPORTANT

The Ignition config files that the installation program generates contain certificates that expire after 24 hours. You must keep the cluster running for 24 hours in a non-degraded state to ensure that the first certificate rotation has finished.



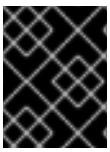
IMPORTANT

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.

1.2.5. Installing the CLI

You can install the CLI in order to interact with OpenShift Container Platform using a command-line interface.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.2. Download and install the new version of **oc**.

Procedure

1. From the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site, navigate to the page for your installation type and click **Download Command-line Tools**.
2. Click the folder for your operating system and architecture and click the compressed file.

NOTE

You can install **oc** on Linux, Windows, or macOS.

3. Save the file to your file system.
4. Extract the compressed file.
5. Place it in a directory that is on your **PATH**.

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

```
$ oc <command>
```

1.2.6. Logging in to the cluster

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

- Deploy an OpenShift Container Platform cluster.
- Install 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:

```
$ oc whoami
system:admin
```

Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

1.3. INSTALLING A CLUSTER ON GCP WITH CUSTOMIZATIONS

In OpenShift Container Platform version 4.2, 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.

Prerequisites

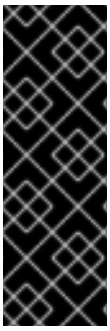
- Review details about the OpenShift Container Platform installation and update processes.
- Configure a GCP account to host the cluster.
- If you use a firewall, you must configure it to allow the sites that your cluster requires access to.

1.3.1. Internet and Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.2, you require access to the internet to install and entitle your cluster. The Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, also requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to the Red Hat OpenShift Cluster Manager . From there, you can allocate entitlements to your cluster.

You must have internet access to:

- Access the Red Hat OpenShift Cluster Manager page to download the installation program and perform subscription management and entitlement. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster. If the Telemetry service cannot entitle your cluster, you must manually entitle it on the Cluster registration page.
- 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.

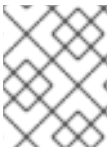
1.3.2. Generating an SSH private key and adding it to the agent

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 to the installation program.

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 rsa -b 4096 -N '' \
    -f <path>/<file_name> 1
```

1 Specify the path and file name, such as **~/.ssh/id_rsa**, of the SSH key.

   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:

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

Agent pid 31874
```

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

```
$ ssh-add <path>/<file_name> (1)

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.

1.3.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on

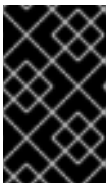
a local computer.

Prerequisites

- You must install the cluster from a computer that uses Linux or macOS.
- You need 500 MB of local disk space to download the installation program.

Procedure

1. Access the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. 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 both the installation program and the files that the installation program creates after you finish installing the cluster.

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

```
$ tar xvf <installation_program>.tar.gz
```

4. From the Pull Secret page on the Red Hat OpenShift Cluster Manager site, download your installation pull secret as a **.txt** file. 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.

1.3.4. Creating the installation configuration file

You can customize your installation of OpenShift Container Platform 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. 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 on, 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.

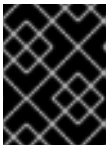
vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

vii. Enter a descriptive name for your cluster. 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.

viii. Paste the pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site.

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.

1.3.4.1. 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

You cannot modify these parameters in the **install-config.yaml** file after installation.

Table 1.4. Required parameters

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. This value 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**.
controlPlane.platform	The cloud provider to host the control plane machines. This parameter value must match the **compute.platform** parameter value.	**aws**, **azure**, **gcp**, **openstack**, or **{}**

Parameter	Description	Values
compute.platform	The cloud provider to host the worker machines. This parameter value must match the **controlPlane.platform** parameter value.	**aws**, **azure**, **gcp**, **openstack**, or **{}**
metadata.name	The name of your cluster.	A string that contains uppercase or lowercase letters, such as **dev**.
platform.<platform>.region	The region to deploy your cluster in.	A valid region for your cloud, such as **us-east-1** for AWS, **centralus** for Azure, or **region1** for Red Hat OpenStack Platform (RHOSP).

pullSecret	The pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site. You use this pull secret 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.	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

Table 1.5. Optional parameters

Parameter	Description	Values
sshKey	The 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 on, specify an SSH key that your **ssh-agent** process uses.	A valid, local public SSH key that you added to the **ssh-agent** process.

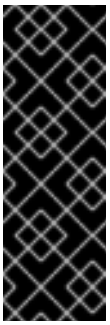
Parameter	Description	Values
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. **IMPORTANT** If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to **2**. The default value is **3**.
controlPlane.hyperthreading	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.	**Enabled** or **Disabled**
controlPlane.replicas	The number of control plane machines to provision.	A positive integer greater than or equal to **3**. The default value is **3**.

Table 1.6. Additional Google Cloud Platform (GCP) parameters

Parameter	Description	Values
platform.gcp.type	The GCP machine type.	The GCP machine type.
platform.gcp.zones	The availability zones where the installation program creates machines for the specified MachinePool.	A list of valid GCP availability zones, such as **us-central1-a**, in a YAML sequence.

1.3.4.2. 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 (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
compute: (5)
- hyperthreading: Enabled (6)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
metadata:
  name: test-cluster (7)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
```

```
    ProjectID: openshift-production 8
    region: us-central-1 9
pullSecret: '{"auths": ...}' 10
sshKey: ssh-ed25519 AAAA... 11
```

1 7 8 9 10 Required. The installation program prompts you for this value.

2 5 If you do not provide these parameters and values, the installation program provides the default value.

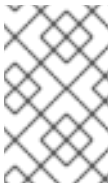
3 6 The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Although 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.

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

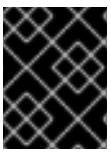


NOTE

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

1.3.5. Deploy 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. Run the installation program:

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

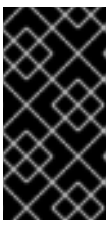
1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

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.



IMPORTANT

The Ignition config files that the installation program generates contain certificates that expire after 24 hours. You must keep the cluster running for 24 hours in a non-degraded state to ensure that the first certificate rotation has finished.

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.

1.3.6. Installing the CLI

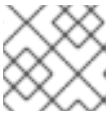
You can install the CLI in order to interact with OpenShift Container Platform using a command-line interface.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.2. Download and install the new version of **oc**.

Procedure

1. From the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site, navigate to the page for your installation type and click **Download Command-line Tools**.
2. Click the folder for your operating system and architecture and click the compressed file.



NOTE

You can install **oc** on Linux, Windows, or macOS.

3. Save the file to your file system.
4. Extract the compressed file.
5. Place it in a directory that is on your **PATH**.

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

```
$ oc <command>
```

1.3.7. Logging in to the cluster

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

- Deploy an OpenShift Container Platform cluster.
- Install 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:

```
$ oc whoami
system:admin
```

Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

1.4. INSTALLING A CLUSTER ON GCP WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.2, 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.

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.

1.4.1. Internet and Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.2, you require access to the internet to install and entitle your cluster. The Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, also requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to the Red Hat OpenShift Cluster Manager . From there, you can allocate entitlements to your cluster.

You must have internet access to:

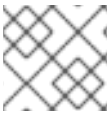
- Access the Red Hat OpenShift Cluster Manager page to download the installation program and perform subscription management and entitlement. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster. If the Telemetry service cannot entitle your cluster, you must manually entitle it on the Cluster registration page.
- 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.

1.4.2. 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 to the installation program.



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 rsa -b 4096 -N '' \
    -f <path>/<file_name> (1)
```

(1) Specify the path and file name, such as **~/.ssh/id_rsa**, of the SSH key.

   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:

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

Agent pid 31874
```

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

```
$ ssh-add <path>/<file_name> (1)

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.

1.4.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on

a local computer.

Prerequisites

- You must install the cluster from a computer that uses Linux or macOS.
- You need 500 MB of local disk space to download the installation program.

Procedure

1. Access the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site. If you have a Red Hat account, log in with your credentials. If you do not, create an account.
2. 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 both the installation program and the files that the installation program creates after you finish installing the cluster.

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

```
$ tar xvf <installation_program>.tar.gz
```

4. From the Pull Secret page on the Red Hat OpenShift Cluster Manager site, download your installation pull secret as a **.txt** file. 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.

1.4.4. Creating the installation configuration file

You can customize your installation of OpenShift Container Platform on Google Cloud Platform (GCP).

Prerequisites

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

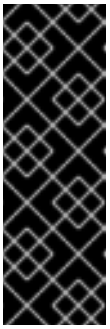
Procedure

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

   a. 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 on, 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.

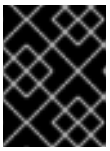
      vi. Select the base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster.

      vii. Enter a descriptive name for your cluster. 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.

      viii. Paste the pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site.

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.

1.4.4.1. 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

You cannot modify these parameters in the **install-config.yaml** file after installation.

Table 1.7. Required parameters

Parameter	Description	Values
baseDomain	The base domain of your cloud provider. This value 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**.
controlPlane.platform	The cloud provider to host the control plane machines. This parameter value must match the **compute.platform** parameter value.	**aws**, **azure**, **gcp**, **openstack**, or **{}**
compute.platform	The cloud provider to host the worker machines. This parameter value must match the **controlPlane.platform** parameter value.	**aws**, **azure**, **gcp**, **openstack**, or **{}**
metadata.name	The name of your cluster.	A string that contains uppercase or lowercase letters, such as **dev**.

Parameter	Description	Values
platform.<platform>.region	The region to deploy your cluster in.	A valid region for your cloud, such as **us-east-1** for AWS, **centralus** for Azure, or **region1** for Red Hat OpenStack Platform (RHOSP).

pullSecret	The pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site. You use this pull secret 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.	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

Table 1.8. Optional parameters

Parameter	Description	Values
sshKey	The 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 on, specify an SSH key that your **ssh-agent** process uses.	A valid, local public SSH key that you added to the **ssh-agent** process.

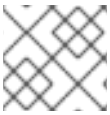
Parameter	Description	Values
compute.hyperthreading	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
compute.replicas	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to **2**. The default value is **3**.
controlPlane.hyperthreading	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.	**Enabled** or **Disabled**
controlPlane.replicas	The number of control plane machines to provision.	A positive integer greater than or equal to **3**. The default value is **3**.

Table 1.9. Additional Google Cloud Platform (GCP) parameters

Parameter	Description	Values
platform.gcp.type	The GCP machine type.	The GCP machine type.
platform.gcp.zones	The availability zones where the installation program creates machines for the specified MachinePool.	A list of valid GCP availability zones, such as **us-central1-a**, in a YAML sequence.

1.4.4.2. Network configuration parameters

You can modify your cluster network configuration parameters in the **install-config.yaml** configuration file. The following table describes the parameters.



NOTE

You cannot modify these parameters in the **install-config.yaml** file after installation.

Table 1.10. Required network parameters

Parameter	Description	Value
networking.networkType	The network plug-in to deploy. The **OpenShiftSDN** plug-in is the only plug-in supported in OpenShift Container Platform 4.2.	The default value is **OpenShiftSDN**.
networking.clusterNetwork.cidr	A block of IP addresses from which Pod IP addresses are allocated. The **OpenShiftSDN** network plug-in supports multiple cluster networks. The address blocks for multiple cluster networks must not overlap. Select address pools large enough to fit your anticipated workload.	An IP address allocation in CIDR format. The default value is **10.128.0.0/14**.
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**, allowing for 510 (2^(32 - 23) - 2) Pod IP addresses.	A subnet prefix. The default value is **23**.
networking.serviceNetwork	A block of IP addresses for services. **OpenShiftSDN** allows only one **serviceNetwork** block. The address block must not overlap with any other network block.	An IP address allocation in CIDR format. The default value is **172.30.0.0/16**.
networking.machineCIDR	A block of IP addresses used by the OpenShift Container Platform installation program while installing the cluster. The address block must not overlap with any other network block.	An IP address allocation in CIDR format. The default value is **10.0.0.0/16**.

1.4.4.3. 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 (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
compute: (5)
- hyperthreading: Enabled (6)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
metadata:
  name: test-cluster (7)
networking: (8)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    ProjectID: openshift-production (9)
    region: us-central-1 (10)
pullSecret: '{"auths": ...}' (11)
sshKey: ssh-ed25519 AAAA... (12)
```

(1) (7) (9) (10) (11) Required. The installation program prompts you for this value.

(2) (5) (8) If you do not provide these parameters and values, the installation program provides the default value.

(3) (6)

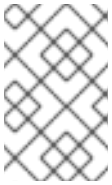
The **controlPlane** section is a single mapping, but the compute section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section

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 **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.

12 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 on, specify an SSH key that your **ssh-agent** process uses.

1.4.5. Modifying advanced network configuration parameters

You can modify the advanced network configuration parameters only before you install the cluster. Advanced configuration customization lets you integrate your cluster into your existing network environment by specifying an MTU or VXLAN port, by allowing customization of kube-proxy settings, and by specifying a different **mode** for the **openshiftSDNConfig** parameter.

IMPORTANT

Modifying the OpenShift Container Platform manifest files directly is not supported.

Prerequisites

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

Procedure

1. Use the following command to create manifests:

```
$ ./openshift-install create manifests --dir=<installation_directory> 1
```

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

2. Modify the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file to prevent Pods from being scheduled on the control plane machines:

   a. Open the **manifests/cluster-scheduler-02-config.yml** file.

b. Locate the **mastersSchedulable** parameter and set its value to **False**.

c. Save and exit the file.

NOTE

Currently, due to a Kubernetes limitation, router Pods running on control plane machines will not be reachable by the ingress load balancer.

3. 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
```

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-*
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

4. Open the **cluster-network-03-config.yml** file in an editor and enter a CR that describes the Operator configuration you want:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec: 1
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: OpenShiftSDN
    openshiftSDNConfig:
      mode: NetworkPolicy
      mtu: 1450
      vxlanPort: 4789
```

1 The parameters for the **spec** parameter are only an example. Specify your configuration for the Cluster Network Operator in the CR.

The CNO provides default values for the parameters in the CR, so you must specify only the parameters that you want to change.

5. Save the **cluster-network-03-config.yml** file and quit the text editor.

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

1.4.6. Cluster Network Operator custom resource (CR)

The cluster network configuration in the **Network.operator.openshift.io** custom resource (CR) stores the configuration settings for the Cluster Network Operator (CNO). The Operator manages the cluster network.

You can specify the cluster network configuration for your OpenShift Container Platform cluster by setting the parameters for the **defaultNetwork** parameter in the CNO CR. The following CR displays the default configuration for the CNO and explains both the parameters you can configure and valid parameter values:

Cluster Network Operator CR

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  clusterNetwork: 1
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork: 2
  - 172.30.0.0/16
  defaultNetwork: 3
    ...
  kubeProxyConfig: 4
    iptablesSyncPeriod: 30s 5
    proxyArguments:
      iptables-min-sync-period: 6
      - 30s
```

1 2 Specified in the **install-config.yaml** file.

3 Configures the software-defined networking (SDN) for the cluster network.

4 The parameters for this object specify the **kube-proxy** configuration. If you do not specify the parameter values, the Network Operator applies the displayed default parameter values.

5 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.

6 The minimum duration before refreshing **iptables** rules. This parameter ensures that the refresh does not happen too frequently. Valid suffixes include **s**, **m**, and **h** and are described in the Go time package

1.4.6.1. Configuration parameters for OpenShift SDN

The following YAML object describes the configuration parameters for OpenShift SDN:

```
defaultNetwork:
```

```
type: OpenShiftSDN (1)
openshiftSDNConfig: (2)
  mode: NetworkPolicy (3)
  mtu: 1450 (4)
  vxlanPort: 4789 (5)
```

1. Specified in the **install-config.yaml** file.
2. Specify only if you want to override part of the OpenShift SDN configuration.
3. Configures the network isolation mode for **OpenShiftSDN**. The allowed values are **Multitenant**, **Subnet**, or **NetworkPolicy**. The default value is **NetworkPolicy**.
4. MTU for the VXLAN overlay network. This value is normally configured automatically, but if the nodes in your cluster do not all use the same MTU, then you must set this explicitly to 50 less than the smallest node MTU value.
5. The port to use for all VXLAN packets. The default value is **4789**. 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 VXLAN, since 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**.

1.4.6.2. Cluster Network Operator example CR

A complete CR for the CNO is displayed in the following example:

Cluster Network Operator example CR

```
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: OpenShiftSDN
    openshiftSDNConfig:
      mode: NetworkPolicy
      mtu: 1450
      vxlanPort: 4789
  kubeProxyConfig:
    iptablesSyncPeriod: 30s
    proxyArguments:
      iptables-min-sync-period:
      - 30s
```

1.4.7. Deploy 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.

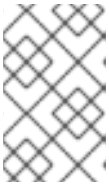
Procedure

1. Run the installation program:

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

1 For **<installation_directory>**, specify the

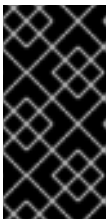
2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.



NOTE

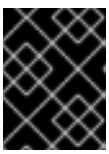
If the cloud provider account that you configured on your host does not have sufficient permissions to deploy the cluster, the installation process stops, and the missing permissions are displayed.

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.



IMPORTANT

The Ignition config files that the installation program generates contain certificates that expire after 24 hours. You must keep the cluster running for 24 hours in a non-degraded state to ensure that the first certificate rotation has finished.

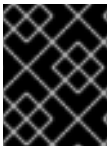


IMPORTANT

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

1.4.8. Installing the CLI

You can install the CLI in order to interact with OpenShift Container Platform using a command-line interface.

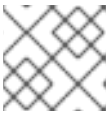


IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.2. Download and install the new version of **oc**.

Procedure

1. From the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site, navigate to the page for your installation type and click **Download Command-line Tools**.
2. Click the folder for your operating system and architecture and click the compressed file.



NOTE

You can install **oc** on Linux, Windows, or macOS.

3. Save the file to your file system.
4. Extract the compressed file.
5. Place it in a directory that is on your **PATH**.

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

```
$ oc <command>
```

1.4.9. Logging in to the cluster

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

- Deploy an OpenShift Container Platform cluster.
- Install 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:

```
$ oc whoami
system:admin
```

Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

1.5. INSTALLING A CLUSTER ON GCP USING DEPLOYMENT MANAGER TEMPLATES

In OpenShift Container Platform version 4.2, 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; the templates are just an example.

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.

NOTE

Be sure to also review this site list if you are configuring a proxy.

1.5.1. 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.

1.5.2. Configuring your GCP project

Before you can install OpenShift Container Platform, you must configure a Google Cloud Platform (GCP) project to host it.

1.5.2.1. 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.

1.5.2.2. 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 1.11. Required API services

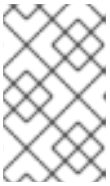
API service	Console service name
Cloud Deployment Manager V2 API	**deploymentmanager.googleapis.com**
Compute Engine API	**compute.googleapis.com**
Google Cloud APIs	**cloudapis.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management (IAM) API	**iam.googleapis.com**
Service Management API	**servicemanagement.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

1.5.2.3. 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.

1.5.2.4. GCP account limits

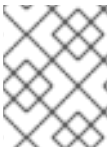
The OpenShift Container Platform cluster uses a number of Google Cloud Platform (GCP) components, but the default Quotas do not affect your ability to install a default OpenShift Container Platform cluster.

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 1.12. GCP resources used in a default cluster

Service	Component	Location	Total resources required	Resources removed after bootstrap
Service account	IAM	Global	5	0
Firewall Rules	Networking	Global	11	1
Forwarding Rules	Compute	Global	2	0
Health checks	Compute	Global	2	0
Images	Compute	Global	1	0
Networks	Networking	Global	1	0

Service	Component	Location	Total resources required	Resources removed after bootstrap
Routers	Networking	Global	1	0
Routes	Networking	Global	2	0
Subnetworks	Compute	Global	2	0
Target Pools	Networking	Global	2	0



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. :!template:

1.5.2.5. 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 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.

1.5.2.5.1. 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:

Required roles for the installation program

- Compute Admin
- DNS Administrator
- 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 1.13. GCP service account permissions

Account	Roles
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

1.5.2.6. 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-south1 (Mumbai, India)
- asia-southeast1 (Jurong West, Singapore)
- 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)

1.5.2.7. 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.

1.5.3. 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.

1.5.3.1. Creating the installation configuration file

You can customize your installation of OpenShift Container Platform on Google Cloud Platform (GCP).

Prerequisites

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

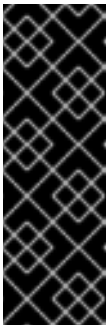
Procedure

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

   a. 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 on, 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. 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.

      viii. Paste the pull secret that you obtained from the Pull Secret page on the Red Hat OpenShift Cluster Manager site.

   c. Optional: If you do not want the cluster to provision compute machines, empty the compute

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
  name: worker
  platform: {}
  replicas: 0 1
```

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.

1.5.3.2. 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

- An existing **install-config.yaml** file.
- Review the sites that your cluster requires access to and determine whether any need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. Add 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 by default with the instance metadata endpoint (**169.254.169.254**) and with the values of the **networking.machineCIDR**, **networking.clusterNetwork.cidr**, and **networking.serviceNetwork** fields from your installation configuration.

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: http://<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. If this field is not specified, then **httpProxy** is used for both HTTP and HTTPS connections. The URL scheme must be **http**; **https** is currently not supported.
3. A comma-separated list of destination domain names, domains, IP addresses, or other network CIDRs to exclude proxying. Preface a domain with **.** to include all subdomains of that domain. Use ***** to bypass proxy for all destinations.
4. If provided, the installation program generates a ConfigMap that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** ConfigMap that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this ConfigMap is referenced in the Proxy object's **trustedCA** field. 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.

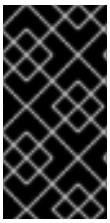
The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster** Proxy object is still created, but it will have a nil **spec**.

NOTE

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

1.5.3.3. 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.



IMPORTANT

The Ignition config files that the installation program generates contain certificates that expire after 24 hours. You must complete your cluster installation and keep the cluster running for 24 hours in a non-degraded state to ensure that the first certificate rotation has finished.

Prerequisites

- Obtain the OpenShift Container Platform installation program.
- Create the **install-config.yaml** installation configuration file.

Procedure

1. Generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir=<installation_directory> 1

WARNING There are no compute nodes specified. The cluster will not fully initialize without
compute nodes.
INFO Consuming "Install Config" from target directory
```

1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

   Because you create your own compute machines later in the installation process, you can safely ignore this warning.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f 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 openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

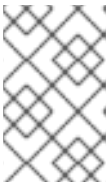
   Because you create and manage the worker machines yourself, you do not need to initialize these machines.

4. Modify the **manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file to prevent Pods from being scheduled on the control plane machines:

   a. Open the **manifests/cluster-scheduler-02-config.yml** file.

   b. Locate the **mastersSchedulable** parameter and set its value to **False**.

   c. Save and exit the file.



NOTE

Currently, due to a Kubernetes limitation, router Pods running on control plane machines will not be reachable by the ingress load balancer. This step might not be required in a future minor version of OpenShift Container Platform.

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 **manifests/cluster-dns-02-config.yml** DNS configuration file:

```
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: {}
```

1 2 Remove these sections completely.

If you do so, you must add ingress DNS records manually in a later step.

6. Obtain the Ignition config files:

```
$ ./openshift-install create ignition-configs --dir=<installation_directory> 1
```

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
```

Additional resources

- Optional: Adding the ingress DNS records

1.5.4. Exporting common variables

1.5.4.1. Extracting the infrastructure name

The Ignition configs contain a unique cluster identifier that you can use to uniquely identify your cluster in Google Cloud Platform (GCP). The provided Deployment Manager templates contain references to this infrastructure name, so you must extract it.

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

- 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
openshift-vw9j6 2
```

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

2 The output of this command is your cluster name and a random string.

1.5.4.2. 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

- 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 1
$ 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`
```

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

1.5.5. 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

resources:
- name: cluster-vpc
 type: 01_vpc.py
 properties:
   infra_id: '${INFRA_ID}' 1
   region: '${REGION}' 2

   master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
   worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
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-east1**.
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:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

1.5.5.1. 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:

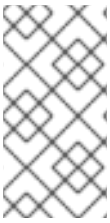
01_vpc.py Deployment Manager template

```
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'] + '-master-nat-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-nat-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        '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': 'MANUAL_ONLY',
            'natIps': ['$(ref.' + context.properties['infra_id'] + '-master-nat-ip.selfLink)'],
            'minPortsPerVm': 7168,
            'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
            'subnetworks': [{
                'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                'sourceIpRangesToNat': ['ALL_IP_RANGES']
            }]
        }, {
            'name': context.properties['infra_id'] + '-nat-worker',
            'natIpAllocateOption': 'MANUAL_ONLY',
            'natIps': ['$(ref.' + context.properties['infra_id'] + '-worker-nat-ip.selfLink)'],
            'minPortsPerVm': 128,
            'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
            'subnetworks': [{
                'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                'sourceIpRangesToNat': ['ALL_IP_RANGES']
            }]
        }]
    }
}]

return {'resources': resources}
```

1.5.6. Creating networking and load balancing components in GCP

You must configure networking and load balancing 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 network and load balancers** section of this topic and save it as **02_infra.py** on your computer. This template describes the networking and load balancing objects that your cluster requires.
2. Export the following variable required by the resource definition:

```
$ export CLUSTER_NETWORK=`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`
```

3. Create a **02_infra.yaml** resource definition file:

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_infra.py

resources:
- name: cluster-infra
  type: 02_infra.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)

    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' (3)
    cluster_network: '${CLUSTER_NETWORK}' (4)
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-east1**.

(3) **cluster_domain** is the domain for the cluster, for example **openshift.example.com**.

(4) **cluster_network** is the **selfLink** URL to the cluster network.

4. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

5. The templates do not create DNS entries due to limitations of Deployment Manager, so you must create them manually:

   a. Export the following variable:

```
$ export CLUSTER_IP=`gcloud compute addresses describe ${INFRA_ID}-cluster-public-ip --region=${REGION} --format json | jq -r .address`
```

   b. Add 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 ${CLUSTER_IP} --name api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

   c. Add 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
```

1.5.6.1. Deployment Manager template for the network and load balancers

You can use the following Deployment Manager template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

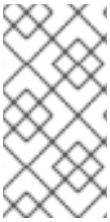
02_infra.py Deployment Manager template

```
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'
        }
    }, {
        'name': context.properties['infra_id'] + '-ign-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 22624,
            'requestPath': '/healthz'
        }
    }, {
```

```
        'name': context.properties['infra_id'] + '-ign-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-ign-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-ign-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-ign-target-pool.selfLink)',
            'portRange': '22623'
        }
    }, {
        '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}
```

1.5.7. Creating firewall rules and IAM roles in GCP

You must create security groups and 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 firewall rules and IAM roles** section of this topic and save it as **03_security.py** on your computer. This template describes the security groups and roles that your cluster requires.

2. Export the following variables required by the resource definition:

```
$ export MASTER_NAT_IP=`gcloud compute addresses describe ${INFRA_ID}-master-nat-ip --region ${REGION} --format json | jq -r .address`
$ export WORKER_NAT_IP=`gcloud compute addresses describe ${INFRA_ID}-worker-nat-ip --region ${REGION} --format json | jq -r .address`
```

3. Create a **03_security.yaml** resource definition file:

```
$ cat <<EOF >03_security.yaml
imports:
- path: 03_security.py

resources:
- name: cluster-security
  type: 03_security.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)

    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
    master_nat_ip: '${MASTER_NAT_IP}' (5)
    worker_nat_ip: '${WORKER_NAT_IP}' (6)
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-east1**.

(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**.

(5) **master_nat_ip** is the IP address of the master NAT, for example **34.94.100.1**.

(6) **worker_nat_ip** is the IP address of the worker NAT, for example **34.94.200.1**.

4. Create the deployment by using the **gcloud** CLI:

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-security --config 03_security.yaml
```

5. The templates do not create the policy bindings due to limitations of Deployment Manager, so you must create them manually:

```
$ export MASTER_SA=${INFRA_ID}-m@${PROJECT_NAME}.iam.gserviceaccount.com
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member "serviceAccount:${MASTER_SA}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
```

```
"serviceAccount:${MASTER_SA}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SA}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SA}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SA}" --role "roles/storage.admin"

$ export WORKER_SA=${INFRA_ID}-w@${PROJECT_NAME}.iam.gserviceaccount.com
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SA}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SA}" --role "roles/storage.admin"
```

6. 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_SA}
```

1.5.7.1. Deployment Manager template for firewall rules and IAM roles

You can use the following Deployment Manager template to deploy the security objects that you need for your OpenShift Container Platform cluster:

03_security.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges':  ['0.0.0.0/0'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-mcs',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceRanges':  [
                context.properties['network_cidr'],
                context.properties['master_nat_ip'],
                context.properties['worker_nat_ip']
            ],
            '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', '22624']
            }],
            'sourceRanges':  ['35.191.0.0/16', '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']
            }],
            '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': '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'
        ]
    }
}, {
    '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}
```

1.5.8. 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 Product Downloads page on the Red Hat customer portal or 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>-gcp.tar**.

2. Export the following variable:

```
$ export IMAGE_SOURCE=<downloaded_image_file_path>
```

3. Create the cluster image:

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

1.5.9. 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.

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 following variables required by the resource definition:

```
$ export CONTROL_SUBNET=`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`
$ export CLUSTER_IMAGE=`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`
$ 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`
```

3. Create a bucket and upload the **bootstrap.ign** file:

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
$ gsutil cp 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: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zone: '${ZONE_0}' (3)

    cluster_network: '${CLUSTER_NETWORK}' (4)
    control_subnet: '${CONTROL_SUBNET}' (5)
    image: '${CLUSTER_IMAGE}' (6)
    machine_type: 'n1-standard-4' (7)
    root_volume_size: '128' (8)

    bootstrap_ign: '${BOOTSTRAP_IGN}' (9)
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-east1**.
3. **zone** is the zone to deploy the bootstrap instance into, for example **us-east1-b**.
4. **cluster_network** is the **selfLink** URL to the cluster network.
5. **control_subnet** is the **selfLink** URL to the control subnet.
6. **image** is the **selfLink** URL to the RHCOS image.
7. **machine_type** is the machine type of the instance, for example **n1-standard-4**.
8. **bootstrap_ign** is the URL output when creating a signed URL above.

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:

```
$ gcloud compute target-pools add-instances \
    ${INFRA_ID}-api-target-pool --instances-zone="${ZONE_0}" --instances=${INFRA_ID}-
bootstrap
$ gcloud compute target-pools add-instances \
    ${INFRA_ID}-ign-target-pool --instances-zone="${ZONE_0}" --instances=${INFRA_ID}-
bootstrap
```

1.5.9.1. Deployment Manager template for the bootstrap machine

You can use the following Deployment Mananger template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

04_bootstrap.py Deployment Manager template

```
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-in-ssh',
    'type': 'compute.v1.firewall',
    'properties': {
        'network': context.properties['cluster_network'],
        'allowed': [{
            'IPProtocol': 'tcp',
            'ports': ['22']
```

```
            }],
            'sourceRanges':  ['0.0.0.0/0'],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    }, {
        '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'] +
'","verification":{}}},"timeouts":{},"version":"2.1.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}',
                }]
            },
            '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']
        }
    }]

    return {'resources': resources}
```

1.5.10. 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 variables needed by the resource definition:

```
$ export MASTER_SERVICE_ACCOUNT_EMAIL=`gcloud iam service-accounts list | grep
"^${INFRA_ID}-master-node " | awk '{print $2}'`
$ export MASTER_IGNITION=`cat 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: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zones: (3)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT_EMAIL}' (7)

    ignition: '${MASTER_IGNITION}' (8)
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-east1**.

3. **zones** are the zones to deploy the bootstrap instance into, for example **us-east1-b**, **us-east1-c**, and **us-east1-d**.
4. **control_subnet** is the **selfLink** URL to the control 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 master service account created above.
8. **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 DNS entries due to limitations of Deployment Manager, so you must add the etcd entries manually:

```
$ export MASTER0_IP=`gcloud compute instances describe ${INFRA_ID}-m-0 --zone
${ZONE_0} --format json | jq -r .networkInterfaces[0].networkIP`
$ export MASTER1_IP=`gcloud compute instances describe ${INFRA_ID}-m-1 --zone
${ZONE_1} --format json | jq -r .networkInterfaces[0].networkIP`
$ export MASTER2_IP=`gcloud compute instances describe ${INFRA_ID}-m-2 --zone
${ZONE_2} --format json | jq -r .networkInterfaces[0].networkIP`
$ 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 ${MASTER0_IP} --name etcd-
0.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-
zone
$ gcloud dns record-sets transaction add ${MASTER1_IP} --name etcd-
1.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-
zone
$ gcloud dns record-sets transaction add ${MASTER2_IP} --name etcd-
2.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-private-
zone
$ gcloud dns record-sets transaction add \
  "0 10 2380 etcd-0.${CLUSTER_NAME}.${BASE_DOMAIN}." \
  "0 10 2380 etcd-1.${CLUSTER_NAME}.${BASE_DOMAIN}." \
  "0 10 2380 etcd-2.${CLUSTER_NAME}.${BASE_DOMAIN}." \
  --name _etcd-server-ssl._tcp.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type SRV -
-zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

6. The templates do not manage load balancer membership due to limitations of Deployment Manager, so you must add the control plane machines manually:

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-m-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-m-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
```

```
zone="${ZONE_2}" --instances=${INFRA_ID}-m-2
$ gcloud compute target-pools add-instances ${INFRA_ID}-ign-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-m-0
$ gcloud compute target-pools add-instances ${INFRA_ID}-ign-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-m-1
$ gcloud compute target-pools add-instances ${INFRA_ID}-ign-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-m-2
```

1.5.10.1. Deployment Manager template for control plane machines

You can use the following Deployment Mananger template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

05_control_plane.py Deployment Manager template

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-m-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'] + '-m-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'] + '-m-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': [{
                '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'][2]
    }
}]

return {'resources': resources}
```

1.5.11. 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:

```
$ ./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:

```
$ gcloud compute target-pools remove-instances ${INFRA_ID}-api-target-pool --instances-
```

```
zone="${ZONE_0}" --instances=${INFRA_ID}-bootstrap
$ gcloud compute target-pools remove-instances ${INFRA_ID}-ign-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-bootstrap
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

1.5.12. 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 following variables needed by the resource definition:

```
$ export COMPUTE_SUBNET=`gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink`
$ export WORKER_SERVICE_ACCOUNT_EMAIL=`gcloud iam service-accounts list | grep
"^${INFRA_ID}-worker-node " | awk '{print $2}'`
$ export WORKER_IGNITION=`cat worker.ign`
```

3. Create a **06_worker.yaml** resource definition file:

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'w-a-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    region: '${REGION}' (3)
    zone: '${ZONE_0}' (4)

    compute_subnet: '${COMPUTE_SUBNET}' (5)
    image: '${CLUSTER_IMAGE}' (6)
    machine_type: 'n1-standard-4' (7)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT_EMAIL}' (8)

    ignition: '${WORKER_IGNITION}' (9)
EOF
```

1. **name** is the name of the worker machine, for example **w-a-0**.
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-east1**.
4. **zone** is the zone to deploy the worker machine into, for example **us-east1-b**.
5. **compute_subnet** is the **selfLink** URL to the compute subnet.
6. **image** is the **selfLink** URL to the RHCOS image.
7. **machine_type** is the machine type of the instance, for example **n1-standard-4**.
8. **service_account_email** is the email address for the worker service account created above.
9. **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
```

1.5.12.1. Deployment Manager template for worker machines

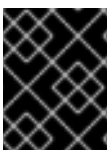
You can use the following Deployment Manager template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

06_worker.py Deployment Manager template

```
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']
        }
    }]

    return {'resources': resources}
```

1.5.13. Installing the CLI

You can install the CLI in order to interact with OpenShift Container Platform using a command-line interface.



IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.2. Download and install the new version of **oc**.

Procedure

1. From the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site, navigate to the page for your installation type and click **Download Command-line Tools**.
2. Click the folder for your operating system and architecture and click the compressed file.

NOTE

You can install **oc** on Linux, Windows, or macOS.

3. Save the file to your file system.
4. Extract the compressed file.
5. Place it in a directory that is on your **PATH**.

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

```
$ oc <command>
```

1.5.14. Logging in to the cluster

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

- Deploy an OpenShift Container Platform cluster.
- Install 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:

```
$ oc whoami
system:admin
```

1.5.15. Approving the CSRs for your machines

When you add machines to a cluster, two pending certificates signing request (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes

NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.14.6+c4799753c
master-1  Ready     master  63m  v1.14.6+c4799753c
master-2  Ready     master  64m  v1.14.6+c4799753c
worker-0  NotReady  worker  76s  v1.14.6+c4799753c
worker-1  NotReady  worker  70s  v1.14.6+c4799753c
```

   The output lists all of the machines that you created.

2. Review the pending certificate signing requests (CSRs) and ensure that the you see a client and server request with **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr

NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending (1)
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending (2)
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

   (1) A client request CSR.

   (2) A server request CSR.

   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. After you approve the initial CSRs, the subsequent node client CSRs are automatically approved by the cluster **kube-controller-manager**. You must implement a method of automatically approving the kubelet serving certificate requests.

- 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
```

1.5.16. 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
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer   172.30.18.154   35.233.157.184
80:32288/TCP,443:31215/TCP   98
```

2. Add the A record to your public and private zones:

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-headers | awk
'{print $4}'`

$ 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}

$ 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
```

If you prefer to add explicit domains instead of using a wildcard, you can 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
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
```

1.5.17. 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 1

INFO Waiting up to 30m0s for the cluster to initialize...
```

1 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. You must keep the cluster running for 24 hours in a non-degraded state to ensure that the first certificate rotation has finished.

2. Observe the running state of your cluster.

   a. Run the following command to view the current cluster version and status:

```
$ oc get clusterversion
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version             False       True          24m     Working towards 4.2.0-0: 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
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.2.0-0   True        False         False      6m18s
cloud-credential                           4.2.0-0   True        False         False      17m
cluster-autoscaler                         4.2.0-0   True        False         False      80s
console                                    4.2.0-0   True        False         False      3m57s
dns                                        4.2.0-0   True        False         False      22m
image-registry                             4.2.0-0   True        False         False      5m4s
ingress                                    4.2.0-0   True        False         False      4m38s
insights                                   4.2.0-0   True        False         False      21m
kube-apiserver                             4.2.0-0   True        False         False      12m
kube-controller-manager                    4.2.0-0   True        False         False      12m
kube-scheduler                             4.2.0-0   True        False         False      11m
machine-api                                4.2.0-0   True        False         False      18m
machine-config                             4.2.0-0   True        False         False      22m
marketplace                                4.2.0-0   True        False         False      5m38s
monitoring                                 4.2.0-0   True        False         False      86s
network                                    4.2.0-0   True        False         False      14m
node-tuning                                4.2.0-0   True        False         False      6m8s
openshift-apiserver                        4.2.0-0   True        False         False      6m48s
openshift-controller-manager               4.2.0-0   True        False         False      12m
openshift-samples                          4.2.0-0   True        False         False      67s
operator-lifecycle-manager                 4.2.0-0   True        False         False      15m
operator-lifecycle-manager-catalog         4.2.0-0   True        False         False      15m
operator-lifecycle-manager-packageserver   4.2.0-0   True        False         False
6m48s
service-ca                                 4.2.0-0   True        False         False      17m
service-catalog-apiserver                  4.2.0-0   True        False         False      6m18s
service-catalog-controller-manager         4.2.0-0   True        False         False      6m19s
storage                                    4.2.0-0   True        False         False      6m20s
```

   c. Run the following command to view your cluster Pods:

```
$ oc get pods --all-namespaces
NAMESPACE                                  NAME
READY   STATUS    RESTARTS   AGE
kube-system                                etcd-member-ip-10-0-3-111.us-east-
2.compute.internal             1/1     Running   0          35m
kube-system                                etcd-member-ip-10-0-3-239.us-east-
2.compute.internal             1/1     Running   0          37m
kube-system                                etcd-member-ip-10-0-3-24.us-east-
2.compute.internal             1/1     Running   0          35m
openshift-apiserver-operator               openshift-apiserver-operator-6d6674f4f4-
```

```
h7t2t                                    1/1     Running     1          37m
openshift-apiserver                      apiserver-fm48r
1/1     Running     0          30m
openshift-apiserver                      apiserver-fxkvv
1/1     Running     0          29m
openshift-apiserver                      apiserver-q85nm
1/1     Running     0          29m
...
openshift-service-ca-operator            openshift-service-ca-operator-66ff6dc6cd-
9r257                    1/1     Running     0          37m
openshift-service-ca                     apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1     Running     0          35m
openshift-service-ca                     configmap-cabundle-injector-8498544d7-
25qn6                    1/1     Running     0          35m
openshift-service-ca                     service-serving-cert-signer-6445fc9c6-wqdqn
1/1     Running     0          35m
openshift-service-catalog-apiserver-operator          openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w      1/1     Running     0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm     1/1     Running     0          31m
```

When the current cluster version is **AVAILABLE**, the installation is complete.

Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

1.6. UNINSTALLING A CLUSTER ON GCP

You can remove a cluster that you deployed to Google Cloud Platform (GCP).

1.6.1. Removing a cluster that uses installer-provisioned infrastructure

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

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 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.