



Red Hat OpenShift Service on AWS 4

Install ROSA with HCP clusters

Installing, accessing, and deleting Red Hat OpenShift Service on AWS (ROSA) clusters.

Red Hat OpenShift Service on AWS 4 Install ROSA with HCP clusters

Installing, accessing, and deleting Red Hat OpenShift Service on AWS (ROSA) clusters.

Legal Notice

Copyright © 2024 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 information on how to install Red Hat OpenShift Service on AWS (ROSA) clusters that use hosted control planes.

Table of Contents

CHAPTER 1. CREATING ROSA WITH HCP CLUSTERS USING THE DEFAULT OPTIONS	3
Considerations regarding auto creation mode	3
1.1. OVERVIEW OF THE DEFAULT CLUSTER SPECIFICATIONS	4
1.2. ROSA WITH HCP PREREQUISITES	5
1.2.1. Creating a Virtual Private Cloud for your ROSA with HCP clusters	5
Creating a Virtual Private Cloud using Terraform	6
Creating a Virtual Private Cloud manually	7
Tagging your subnets	7
1.2.2. Creating the account-wide STS roles and policies	9
1.2.3. Creating an OpenID Connect configuration	9
1.2.4. Creating Operator roles and policies	11
1.3. CREATING A ROSA WITH HCP CLUSTER USING THE CLI	13
1.4. NEXT STEPS	15
1.5. ADDITIONAL RESOURCES	16
CHAPTER 2. CREATING ROSA WITH HCP CLUSTERS USING A CUSTOM AWS KMS ENCRYPTION KEY ..	17
2.1. ROSA WITH HCP PREREQUISITES	17
2.1.1. Creating a Virtual Private Cloud for your ROSA with HCP clusters	17
Creating a Virtual Private Cloud using Terraform	17
Creating a Virtual Private Cloud manually	18
2.1.2. Creating the account-wide STS roles and policies	19
2.1.3. Creating an OpenID Connect configuration	20
2.1.4. Creating Operator roles and policies	21
2.1.5. Creating a ROSA cluster using a custom AWS KMS key	24
2.2. NEXT STEPS	27
2.3. ADDITIONAL RESOURCES	27
CHAPTER 3. CREATING A PRIVATE CLUSTER ON ROSA WITH HCP	28
3.1. CREATING AN AWS PRIVATE CLUSTER	28
3.2. CONFIGURING AWS SECURITY GROUPS TO ACCESS THE API	29
3.3. NEXT STEPS	30
3.4. ADDITIONAL RESOURCES	30
CHAPTER 4. USING THE NODE TUNING OPERATOR ON ROSA WITH HCP CLUSTERS	31
Purpose	31
4.1. CUSTOM TUNING SPECIFICATION	31
4.2. CREATING NODE TUNING CONFIGURATIONS ON ROSA WITH HCP	36
4.3. MODIFYING YOUR NODE TUNING CONFIGURATIONS FOR ROSA WITH HCP	38
4.4. DELETING NODE TUNING CONFIGURATIONS ON ROSA WITH HCP	40

CHAPTER 1. CREATING ROSA WITH HCP CLUSTERS USING THE DEFAULT OPTIONS



NOTE

If you are looking for a quickstart guide for ROSA Classic, see [Red Hat OpenShift Service on AWS quickstart guide](#).

Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) offers a more efficient and reliable architecture for creating Red Hat OpenShift Service on AWS (ROSA) clusters. With ROSA with HCP, each cluster has a dedicated control plane that is isolated in a ROSA service account.

Create a ROSA with HCP cluster quickly by using the default options and automatic AWS Identity and Access Management (IAM) resource creation. You can deploy your cluster by using the ROSA CLI (**rosa**).



IMPORTANT

Since it is not possible to upgrade or convert existing ROSA clusters to a hosted control planes architecture, you must create a new cluster to use ROSA with HCP functionality.



NOTE

ROSA with HCP clusters only support AWS Security Token Service (STS) authentication.

Further reading

- For a comparison between ROSA with HCP and ROSA Classic, see the [Comparing architecture models](#) documentation.
- See the AWS documentation for information about [Getting started with ROSA with HCP using the ROSA CLI in auto mode](#).

Additional resources

For a full list of the supported certificates, see the [Compliance](#) section of "Understanding process and security for Red Hat OpenShift Service on AWS".

Considerations regarding auto creation mode

The procedures in this document use the **auto** mode in the ROSA CLI to immediately create the required IAM resources using the current AWS account. The required resources include the account-wide IAM roles and policies, cluster-specific Operator roles and policies, and OpenID Connect (OIDC) identity provider.

Alternatively, you can use **manual** mode, which outputs the **aws** commands needed to create the IAM resources instead of deploying them automatically. For steps to deploy a ROSA with HCP cluster by using **manual** mode or with customizations, see [Creating a cluster using customizations](#).

Next steps



- Ensure that you have completed the [AWS prerequisites](#).

1.1. OVERVIEW OF THE DEFAULT CLUSTER SPECIFICATIONS

You can quickly create a ROSA with HCP cluster with the AWS Security Token Service (STS) by using the default installation options. The following summary describes the default cluster specifications.

Table 1.1. Default ROSA with HCP cluster specifications

Component	Default specifications
Accounts and roles	<ul style="list-style-type: none"> ● Default IAM role prefix: ManagedOpenShift ● No cluster admin role created
Cluster settings	<ul style="list-style-type: none"> ● Default cluster version: Latest ● Default AWS region for installations using the ROSA CLI (rosa): Defined by your aws CLI configuration ● Default EC2 IMDS endpoints (both v1 and v2) are enabled ● Availability: Single zone for the data plane ● Monitoring for user-defined projects: Enabled
Encryption	<ul style="list-style-type: none"> ● Cloud storage is encrypted at rest ● Additional etcd encryption is not enabled ● The default AWS Key Management Service (KMS) key is used as the encryption key for persistent data
Compute node machine pool	<ul style="list-style-type: none"> ● Compute node instance type: m5.xlarge (4 vCPU 16, GiB RAM) ● Compute node count: 2 ● Autoscaling: Not enabled ● No additional node labels
Networking configuration	<ul style="list-style-type: none"> ● Cluster privacy: Public ● You must have configured your own Virtual Private Cloud (VPC) ● No cluster-wide proxy is configured

Component	Default specifications
Classless Inter-Domain Routing (CIDR) ranges	<ul style="list-style-type: none"> ● Machine CIDR: 10.0.0.0/16 ● Service CIDR: 172.30.0.0/16 ● Pod CIDR: 10.128.0.0/16 ● Host prefix: /23 <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>NOTE</p> <p>When using ROSA with HCP, the static IP address 172.20.0.1 is reserved for the internal Kubernetes API address. The machine, pod, and service CIDRs ranges must not conflict with this IP address.</p> </div> </div>
Cluster roles and policies	<ul style="list-style-type: none"> ● Mode used to create the Operator roles and the OpenID Connect (OIDC) provider: auto <div style="display: flex; align-items: flex-start; margin-top: 10px;">  <div> <p>NOTE</p> <p>For installations that use OpenShift Cluster Manager on the Hybrid Cloud Console, the auto mode requires an admin-privileged OpenShift Cluster Manager role.</p> </div> </div> <ul style="list-style-type: none"> ● Default Operator role prefix: <cluster_name>-<4_digit_random_string>
Cluster update strategy	<ul style="list-style-type: none"> ● Individual updates ● 1 hour grace period for node draining

1.2. ROSA WITH HCP PREREQUISITES

To create a ROSA with HCP cluster, you must have the following items:

- A configured virtual private cloud (VPC)
- Account-wide roles
- An OIDC configuration
- Operator roles

1.2.1. Creating a Virtual Private Cloud for your ROSA with HCP clusters

You must have a Virtual Private Cloud (VPC) to create ROSA with HCP cluster. You can use the following methods to create a VPC:

- Create a VPC by using a Terraform template
- Manually create the VPC resources in the AWS console



NOTE

The Terraform instructions are for testing and demonstration purposes. Your own installation requires some modifications to the VPC for your own use. You should also ensure that when you use this Terraform script it is in the same region that you intend to install your cluster. In these examples, use **us-east-2**.

Creating a Virtual Private Cloud using Terraform

Terraform is a tool that allows you to create various resources using an established template. The following process uses the default options as required to create a ROSA with HCP cluster. For more information about using Terraform, see the additional resources.

Prerequisites

- You have installed Terraform version 1.4.0 or newer on your machine.
- You have installed Git on your machine.

Procedure

1. Open a shell prompt and clone the Terraform VPC repository by running the following command:

```
$ git clone https://github.com/openshift-cs/terraform-vpc-example
```

2. Navigate to the created directory by running the following command:

```
$ cd terraform-vpc-example
```

3. Initiate the Terraform file by running the following command:

```
$ terraform init
```

A message confirming the initialization appears when this process completes.

4. To build your VPC Terraform plan based on the existing Terraform template, run the **plan** command. You must include your AWS region. You can choose to specify a cluster name. A **rosa.tfplan** file is added to the **hypershift-tf** directory after the **terraform plan** completes. For more detailed options, see the [Terraform VPC repository's README file](#).

```
$ terraform plan -out rosa.tfplan -var region=<region> [-var cluster_name=<cluster_name>]
```

5. Apply this plan file to build your VPC by running the following command:

```
$ terraform apply rosa.tfplan
```

6. Optional: You can capture the values of the Terraform-provisioned private, public, and machinepool subnet IDs as environment variables to use when creating your ROSA with HCP cluster by running the following commands:

```
$ export SUBNET_IDS=$(terraform output -raw cluster-subnets-string)
```

Verification

- You can verify that the variables were correctly set with the following command:

```
$ echo $SUBNET_IDS
```

Sample output

```
$ subnet-0a6a57e0f784171aa,subnet-078e84e5b10ecf5b0
```

Additional resources

- See the [Terraform VPC](#) repository for a detailed list of all options available when customizing the VPC for your needs.

Creating a Virtual Private Cloud manually

If you choose to manually create your Virtual Private Cloud (VPC) instead of using Terraform, go to [the VPC page in the AWS console](#). Your VPC must meet the requirements shown in the following table.

Table 1.2. Requirements for your VPC

Requirement	Details
VPC name	You need to have the specific VPC name and ID when creating your cluster.
CIDR range	Your VPC CIDR range should match your machine CIDR.
Availability zone	You need one availability zone for a single zone, and you need three for availability zones for multi-zone.
Public subnet	You must have one public subnet with a NAT gateway for public clusters. Private clusters do not need a public subnet.
DNS hostname and resolution	You must ensure that the DNS hostname and resolution are enabled.

Tagging your subnets

Before you can use your VPC to create a ROSA with HCP cluster, you must tag your VPC subnets. Automated service preflight checks verify that these resources are tagged correctly before you can use these resources. The following table shows how your resources should be tagged as the following:

Resource	Key	Value
Public subnet	kubernetes.io/role/elb	1 or no value

Resource	Key	Value
Private subnet	kubernetes.io/role/internal-elb	1 or no value

**NOTE**

You must tag at least one private subnet and, if applicable, and one public subnet.

Prerequisites

- You have created a VPC.
- You have installed the **aws** CLI.

Procedure

1. Verify the tags currently on your subnet by running the following command:

```
$ aws ec2 describe-tags --filters "Name=resource-id,Values=<subnet-id>"
```

Example output

```
TAGS Name <subnet-id> subnet <prefix>-subnet-public1-us-east-1a
```

2. Tag your resources in your terminal by running the following commands:
 - a. For public subnets, run:

```
$ aws ec2 create-tags --resources <public-subnet-id> --tags
Key=kubernetes.io/role/elb,Value=1
```

- b. For private subnets, run:

```
$ aws ec2 create-tags --resources <private-subnet-id> --tags
Key=kubernetes.io/role/internal-elb,Value=1
```

Verification

1. Verify that the tag is correctly applied by running the following command:

```
$ aws ec2 describe-tags --filters "Name=resource-id,Values=<subnet_id>"
```

Example output

```
TAGS Name <subnet-id> subnet <prefix>-subnet-public1-us-east-1a
TAGS kubernetes.io/role/elb <subnet-id> subnet 1
```

Additional resources

- [Get Started with Amazon VPC](#)

- [HashiCorp Terraform documentation](#)
- [Subnet Auto Discovery](#)

1.2.2. Creating the account-wide STS roles and policies

Before using the Red Hat OpenShift Service on AWS (ROSA) CLI (**rosa**) to create Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) clusters, create the required account-wide roles and policies, including the Operator policies.



NOTE

ROSA with HCP clusters require account and Operator roles with AWS managed policies attached. Customer managed policies are not supported. For more information regarding AWS managed policies for ROSA with HCP clusters, see [AWS managed policies for ROSA account roles](#).

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- You have installed and configured the latest ROSA CLI (**rosa**) on your installation host.
- You have logged in to your Red Hat account by using the ROSA CLI.

Procedure

- If they do not exist in your AWS account, create the required account-wide STS roles and attach the policies by running the following command:

```
$ rosa create account-roles --hosted-cp --mode auto --yes
```

- Optional: Set your prefix as an environmental variable by running the following command:

```
$ export ACCOUNT_ROLES_PREFIX=<account_role_prefix>
```

- View the value of the variable by running the following command:

```
$ echo $ACCOUNT_ROLES_PREFIX
```

Example output

```
ManagedOpenShift
```

For more information regarding AWS managed IAM policies for ROSA, see [AWS managed IAM policies for ROSA](#).

1.2.3. Creating an OpenID Connect configuration

When using a ROSA with HCP cluster, you must create the OpenID Connect (OIDC) configuration prior to creating your cluster. This configuration is registered to be used with OpenShift Cluster Manager.

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have completed the AWS prerequisites for Red Hat OpenShift Service on AWS.
- You have installed and configured the latest Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**, on your installation host.

Procedure

- To create your OIDC configuration alongside the AWS resources, run the following command:

```
$ rosa create oidc-config --mode=auto --yes
```

This command returns the following information.

Example output

```
? Would you like to create a Managed (Red Hat hosted) OIDC Configuration Yes
I: Setting up managed OIDC configuration
I: To create Operator Roles for this OIDC Configuration, run the following command and
remember to replace <user-defined> with a prefix of your choice:
  rosa create operator-roles --prefix <user-defined> --oidc-config-id 13cdr6b
If you are going to create a Hosted Control Plane cluster please include '--hosted-cp'
I: Creating OIDC provider using 'arn:aws:iam::4540112244:user/userName'
? Create the OIDC provider? Yes
I: Created OIDC provider with ARN 'arn:aws:iam::4540112244:oidc-
provider/dvbwgdztaeq9o.cloudfront.net/13cdr6b'
```

When creating your cluster, you must supply the OIDC config ID. The CLI output provides this value for **--mode auto**, otherwise you must determine these values based on **aws** CLI output for **--mode manual**.

- Optional: you can save the OIDC configuration ID as a variable to use later. Run the following command to save the variable:

```
$ export OIDC_ID=<oidc_config_id> 1
```

- 1** In the example output above, the OIDC configuration ID is 13cdr6b.

- View the value of the variable by running the following command:

```
$ echo $OIDC_ID
```

Example output

```
13cdr6b
```

Verification

- You can list the possible OIDC configurations available for your clusters that are associated with your user organization. Run the following command:

```
$ rosa list oidc-config
```

Example output

```
ID                MANAGED ISSUER URL
SECRET ARN
2330dbs0n8m3chkk25gkkcd8pnj3lk2 true
https://dvbwgdztaeq9o.cloudfront.net/2330dbs0n8m3chkk25gkkcd8pnj3lk2
233hvnjrjoqu14jltk6lhbhf2tj11f8un false https://oidc-r7u1.s3.us-east-1.amazonaws.com
aws:secretsmanager:us-east-1:242819244:secret:rosa-private-key-oidc-r7u1-tM3MDN
```

1.2.4. Creating Operator roles and policies

When using a ROSA with HCP cluster, you must create the Operator IAM roles that are required for Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) deployments. The cluster Operators use the Operator roles to obtain the temporary permissions required to carry out cluster operations, such as managing back-end storage, cloud provider credentials, and external access to a cluster.

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have installed and configured the latest Red Hat OpenShift Service on AWS ROSA CLI (**rosa**), on your installation host.
- You created the account-wide AWS roles.

Procedure

- Set your prefix name to an environment variable using the following command:

```
$ export OPERATOR_ROLES_PREFIX=<prefix_name>
```

- To create your Operator roles, run the following command:

```
$ rosa create operator-roles --hosted-cp --prefix=$OPERATOR_ROLES_PREFIX --oidc-
config-id=$OIDC_ID --installer-role-arn
arn:aws:iam::${AWS_ACCOUNT_ID}:role/${ACCOUNT_ROLES_PREFIX}-HCP-ROSA-
Installer-Role
```

The following breakdown provides options for the Operator role creation.

```
$ rosa create operator-roles --hosted-cp
--prefix=$OPERATOR_ROLES_PREFIX 1
--oidc-config-id=$OIDC_ID 2
--installer-role-arn
arn:aws:iam::${AWS_ACCOUNT_ID}:role/${ACCOUNT_ROLES_PREFIX}-HCP-ROSA-
Installer-Role 3
```

- 1 You must supply a prefix when creating these Operator roles. Failing to do so produces an error. See the Additional resources of this section for information on the Operator prefix.
- 2 This value is the OIDC configuration ID that you created for your ROSA with HCP cluster.
- 3 This value is the installer role ARN that you created when you created the ROSA account roles.

You must include the **--hosted-cp** parameter to create the correct roles for ROSA with HCP clusters. This command returns the following information.

Sample output

```
? Role creation mode: auto
? Operator roles prefix: <pre-filled_prefix> 1
? OIDC Configuration ID: 23soa2bgvpek9kmes9s7os0a39i13qm4 |
https://dvwbgdztaeq9o.cloudfront.net/23soa2bgvpek9kmes9s7os0a39i13qm4 2
? Create hosted control plane operator roles: Yes
W: More than one Installer role found
? Installer role ARN: arn:aws:iam::4540112244:role/<prefix>-HCP-ROSA-Installer-Role
? Permissions boundary ARN (optional):
I: Reusable OIDC Configuration detected. Validating trusted relationships to operator roles:
I: Creating roles using 'arn:aws:iam::4540112244:user/<userName>'
I: Created role '<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials'
I: Created role '<prefix>-openshift-cloud-network-config-controller-cloud-credenti' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-cloud-network-config-controller-cloud-
credenti'
I: Created role '<prefix>-kube-system-kube-controller-manager' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-kube-controller-manager'
I: Created role '<prefix>-kube-system-capa-controller-manager' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-capa-controller-manager'
I: Created role '<prefix>-kube-system-control-plane-operator' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-control-plane-operator'
I: Created role '<prefix>-kube-system-kms-provider' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-kms-provider'
I: Created role '<prefix>-openshift-image-registry-installer-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-image-registry-installer-cloud-credentials'
I: Created role '<prefix>-openshift-ingress-operator-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-ingress-operator-cloud-credentials'
I: To create a cluster with these roles, run the following command:
rosa create cluster --sts --oidc-config-id 23soa2bgvpek9kmes9s7os0a39i13qm4 --operator-
roles-prefix <prefix> --hosted-cp
```

- 1 This field is prepopulated with the prefix that you set in the initial creation command.
- 2 This field requires you to select an OIDC configuration that you created for your ROSA with HCP cluster.

The Operator roles are now created and ready to use for creating your ROSA with HCP cluster.

Verification

- You can list the Operator roles associated with your ROSA account. Run the following command:

```
$ rosa list operator-roles
```

Sample output

```
I: Fetching operator roles
ROLE PREFIX AMOUNT IN BUNDLE
<prefix> 8
? Would you like to detail a specific prefix Yes 1
? Operator Role Prefix: <prefix>
ROLE NAME ROLE ARN
VERSION MANAGED
<prefix>-kube-system-capac-controller-manager
arn:aws:iam::4540112244:role/<prefix>-kube-system-capac-controller-manager
4.13 No
<prefix>-kube-system-control-plane-operator
arn:aws:iam::4540112244:role/<prefix>-kube-system-control-plane-operator
4.13 No
<prefix>-kube-system-kms-provider
arn:aws:iam::4540112244:role/<prefix>-kube-system-kms-provider 4.13
No
<prefix>-kube-system-kube-controller-manager
arn:aws:iam::4540112244:role/<prefix>-kube-system-kube-controller-manager
4.13 No
<prefix>-openshift-cloud-network-config-controller-cloud-credenti
arn:aws:iam::4540112244:role/<prefix>-openshift-cloud-network-config-controller-cloud-
credenti 4.13 No
<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials
arn:aws:iam::4540112244:role/<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials
4.13 No
<prefix>-openshift-image-registry-installer-cloud-credentials
arn:aws:iam::4540112244:role/<prefix>-openshift-image-registry-installer-cloud-credentials
4.13 No
<prefix>-openshift-ingress-operator-cloud-credentials
arn:aws:iam::4540112244:role/<prefix>-openshift-ingress-operator-cloud-credentials
4.13 No
```

- After the command runs, it displays all the prefixes associated with your AWS account and notes how many roles are associated with this prefix. If you need to see all of these roles and their details, enter "Yes" on the detail prompt to have these roles listed out with specifics.

Additional resources

- See [About custom Operator IAM role prefixes](#) for information on the Operator prefixes.

1.3. CREATING A ROSA WITH HCP CLUSTER USING THE CLI

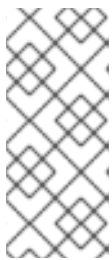
When using the Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**, to create a cluster, you can select the default options to create the cluster quickly.

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- You have installed and configured the latest ROSA CLI (**rosa**) on your installation host. Run **rosa version** to see your currently installed version of the ROSA CLI. If a newer version is available, the CLI provides a link to download this upgrade.
- You have logged in to your Red Hat account by using the ROSA CLI.
- You have created an OIDC configuration.
- You have verified that the AWS Elastic Load Balancing (ELB) service role exists in your AWS account.

Procedure

1. Use one of the following commands to create your ROSA with HCP cluster:



NOTE

When creating a ROSA with HCP cluster, the default machine Classless Inter-Domain Routing (CIDR) is **10.0.0.0/16**. If this does not correspond to the CIDR range for your VPC subnets, add **--machine-cidr <address_block>** to the following commands. To learn more about the default CIDR ranges for Red Hat OpenShift Service on AWS, see [CIDR range definitions](#).

- If you did not set environmental variables, run the following command:

```
$ rosa create cluster --cluster-name=<cluster_name> \ <.>
--mode=auto --hosted-cp [--private] \ <.>
--operator-roles-prefix <operator-role-prefix> \ <.>
--oidc-config-id <id-of-oidc-configuration> \
--subnet-ids=<public-subnet-id>,<private-subnet-id>
```

<.> Specify the name of your cluster. If your cluster name is longer than 15 characters, it will contain an autogenerated domain prefix as a subdomain for your provisioned cluster on openshiftapps.com. To customize the subdomain, use the **--domain-prefix** flag. The domain prefix cannot be longer than 15 characters, must be unique, and cannot be changed after cluster creation. <.> Optional: The **--private** argument is used to create private ROSA with HCP clusters. If you use this argument, ensure that you only use your private subnet ID for **--subnet-ids**. <.> By default, the cluster-specific Operator role names are prefixed with the cluster name and a random 4-digit hash. You can optionally specify a custom prefix to replace **<cluster_name>-<hash>** in the role names. The prefix is applied when you create the cluster-specific Operator IAM roles. For information about the prefix, see *About custom Operator IAM role prefixes*.

**NOTE**

If you specified custom ARN paths when you created the associated account-wide roles, the custom path is automatically detected. The custom path is applied to the cluster-specific Operator roles when you create them in a later step.

- If you set the environmental variables, create a cluster with a single, initial machine pool, using either a publicly or privately available API, and a publicly or privately available Ingress by running the following command:

```
$ rosa create cluster --private --cluster-name=<cluster_name> \
  --mode=auto --hosted-cp --operator-roles-prefix=$OPERATOR_ROLES_PREFIX \
  --oidc-config-id=$OIDC_ID --subnet-ids=$SUBNET_IDS
```

- If you set the environmental variables, create a cluster with a single, initial machine pool, a publicly available API, and a publicly available Ingress by running the following command:

```
$ rosa create cluster --cluster-name=<cluster_name> --mode=auto --hosted-cp --
  operator-roles-prefix=$OPERATOR_ROLES_PREFIX --oidc-config-id=$OIDC_CONFIG
  --subnet-ids=$SUBNET_IDS
```

2. Check the status of your cluster by running the following command:

```
$ rosa describe cluster --cluster=<cluster_name>
```

The following **State** field changes are listed in the output as the cluster installation progresses:

- **pending (Preparing account)**
- **installing (DNS setup in progress)**
- **installing**
- **ready**

**NOTE**

If the installation fails or the **State** field does not change to **ready** after more than 10 minutes, check the installation troubleshooting documentation for details. For more information, see *Troubleshooting installations*. For steps to contact Red Hat Support for assistance, see *Getting support for Red Hat OpenShift Service on AWS*.

3. Track the progress of the cluster creation by watching the Red Hat OpenShift Service on AWS installation program logs. To check the logs, run the following command:

```
$ rosa logs install --cluster=<cluster_name> --watch \ <.>
```

<.> Optional: To watch for new log messages as the installation progresses, use the **--watch** argument.

1.4. NEXT STEPS

- [Accessing a ROSA cluster](#)

1.5. ADDITIONAL RESOURCES

- For steps to deploy a ROSA cluster using manual mode, see [Creating a cluster using customizations](#).
- For more information about the AWS Identity Access Management (IAM) resources required to deploy Red Hat OpenShift Service on AWS with STS, see [About IAM resources for clusters that use STS](#).
- See [Additional custom security groups](#) for information about security group requirements.
- For details about optionally setting an Operator role name prefix, see [About custom Operator IAM role prefixes](#).
- For information about the prerequisites to installing ROSA with STS, see [AWS prerequisites for ROSA with STS](#).
- For details about using the **auto** and **manual** modes to create the required STS resources, see [Understanding the auto and manual deployment modes](#).
- For more information about using OpenID Connect (OIDC) identity providers in AWS IAM, see [Creating OpenID Connect \(OIDC\) identity providers](#) in the AWS documentation.
- For more information about troubleshooting ROSA cluster installations, see [Troubleshooting installations](#).
- For steps to contact Red Hat Support for assistance, see [Getting support for Red Hat OpenShift Service on AWS](#).

CHAPTER 2. CREATING ROSA WITH HCP CLUSTERS USING A CUSTOM AWS KMS ENCRYPTION KEY

Create a Red Hat OpenShift Service on AWS (ROSA) with a hosted control planes (HCP) cluster using a custom AWS Key Management Service (KMS) key.

2.1. ROSA WITH HCP PREREQUISITES

To create a ROSA with HCP cluster, you must have the following items:

- A configured virtual private cloud (VPC)
- Account-wide roles
- An OIDC configuration
- Operator roles

2.1.1. Creating a Virtual Private Cloud for your ROSA with HCP clusters

You must have a Virtual Private Cloud (VPC) to create ROSA with HCP cluster. You can use the following methods to create a VPC:

- Create a VPC by using a Terraform template
- Manually create the VPC resources in the AWS console



NOTE

The Terraform instructions are for testing and demonstration purposes. Your own installation requires some modifications to the VPC for your own use. You should also ensure that when you use this Terraform script it is in the same region that you intend to install your cluster. In these examples, use **us-east-2**.

Creating a Virtual Private Cloud using Terraform

Terraform is a tool that allows you to create various resources using an established template. The following process uses the default options as required to create a ROSA with HCP cluster. For more information about using Terraform, see the additional resources.

Prerequisites

- You have installed Terraform version 1.4.0 or newer on your machine.
- You have installed Git on your machine.

Procedure

1. Open a shell prompt and clone the Terraform VPC repository by running the following command:

```
$ git clone https://github.com/openshift-cs/terraform-vpc-example
```

2. Navigate to the created directory by running the following command:

```
$ cd terraform-vpc-example
```

- Initiate the Terraform file by running the following command:

```
$ terraform init
```

A message confirming the initialization appears when this process completes.

- To build your VPC Terraform plan based on the existing Terraform template, run the **plan** command. You must include your AWS region. You can choose to specify a cluster name. A **rosa.tfplan** file is added to the **hypershift-tf** directory after the **terraform plan** completes. For more detailed options, see the [Terraform VPC repository's README file](#).

```
$ terraform plan -out rosa.tfplan -var region=<region> [-var cluster_name=<cluster_name>]
```

- Apply this plan file to build your VPC by running the following command:

```
$ terraform apply rosa.tfplan
```

- Optional: You can capture the values of the Terraform-provisioned private, public, and machinepool subnet IDs as environment variables to use when creating your ROSA with HCP cluster by running the following commands:

```
$ export SUBNET_IDS=$(terraform output -raw cluster-subnets-string)
```

Verification

- You can verify that the variables were correctly set with the following command:

```
$ echo $SUBNET_IDS
```

Sample output

```
$ subnet-0a6a57e0f784171aa,subnet-078e84e5b10ecf5b0
```

Additional resources

- See the [Terraform VPC](#) repository for a detailed list of all options available when customizing the VPC for your needs.

Creating a Virtual Private Cloud manually

If you choose to manually create your Virtual Private Cloud (VPC) instead of using Terraform, go to [the VPC page in the AWS console](#). Your VPC must meet the requirements shown in the following table.

Table 2.1. Requirements for your VPC

Requirement	Details
VPC name	You need to have the specific VPC name and ID when creating your cluster.

Requirement	Details
CIDR range	Your VPC CIDR range should match your machine CIDR.
Availability zone	You need one availability zone for a single zone, and you need three for availability zones for multi-zone.
Public subnet	You must have one public subnet with a NAT gateway for public clusters. Private clusters do not need a public subnet.
DNS hostname and resolution	You must ensure that the DNS hostname and resolution are enabled.

Additional resources

- [Get Started with Amazon VPC](#)
- [HashiCorp Terraform documentation](#)

2.1.2. Creating the account-wide STS roles and policies

Before using the Red Hat OpenShift Service on AWS (ROSA) CLI (**rosa**) to create Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) clusters, create the required account-wide roles and policies, including the Operator policies.



NOTE

ROSA with HCP clusters require account and Operator roles with AWS managed policies attached. Customer managed policies are not supported. For more information regarding AWS managed policies for ROSA with HCP clusters, see [AWS managed policies for ROSA account roles](#).

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- You have installed and configured the latest ROSA CLI (**rosa**) on your installation host.
- You have logged in to your Red Hat account by using the ROSA CLI.

Procedure

- If they do not exist in your AWS account, create the required account-wide STS roles and attach the policies by running the following command:

```
$ rosa create account-roles --hosted-cp --mode auto --yes
```

- Optional: Set your prefix as an environmental variable by running the following command:

```
$ export ACCOUNT_ROLES_PREFIX=<account_role_prefix>
```

- View the value of the variable by running the following command:

```
$ echo $ACCOUNT_ROLES_PREFIX
```

Example output

```
ManagedOpenShift
```

For more information regarding AWS managed IAM policies for ROSA, see [AWS managed IAM policies for ROSA](#).

2.1.3. Creating an OpenID Connect configuration

When using a ROSA with HCP cluster, you must create the OpenID Connect (OIDC) configuration prior to creating your cluster. This configuration is registered to be used with OpenShift Cluster Manager.

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have completed the AWS prerequisites for Red Hat OpenShift Service on AWS.
- You have installed and configured the latest Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**, on your installation host.

Procedure

- To create your OIDC configuration alongside the AWS resources, run the following command:

```
$ rosa create oidc-config --mode=auto --yes
```

This command returns the following information.

Example output

```
? Would you like to create a Managed (Red Hat hosted) OIDC Configuration Yes
I: Setting up managed OIDC configuration
I: To create Operator Roles for this OIDC Configuration, run the following command and
remember to replace <user-defined> with a prefix of your choice:
  rosa create operator-roles --prefix <user-defined> --oidc-config-id 13cdr6b
If you are going to create a Hosted Control Plane cluster please include '--hosted-cp'
I: Creating OIDC provider using 'arn:aws:iam::4540112244:user/userName'
? Create the OIDC provider? Yes
I: Created OIDC provider with ARN 'arn:aws:iam::4540112244:oidc-
provider/dvbwgdztaeq9o.cloudfront.net/13cdr6b'
```


When creating your cluster, you must supply the OIDC config ID. The CLI output provides this value for **--mode auto**, otherwise you must determine these values based on **aws** CLI output for **--mode manual**.

- Optional: you can save the OIDC configuration ID as a variable to use later. Run the following command to save the variable:

```
$ export OIDC_ID=<oidc_config_id> 1
```

- 1** In the example output above, the OIDC configuration ID is 13cdr6b.

- View the value of the variable by running the following command:

```
$ echo $OIDC_ID
```

Example output

```
13cdr6b
```

Verification

- You can list the possible OIDC configurations available for your clusters that are associated with your user organization. Run the following command:

```
$ rosa list oidc-config
```

Example output

```
ID                MANAGED ISSUER URL
SECRET ARN
2330db0n8m3chkkr25gkkcd8pnj3lk2 true
https://dvbwgdztaeq9o.cloudfront.net/2330db0n8m3chkkr25gkkcd8pnj3lk2
233hvnrjoqu14jltk6lhbhf2tj11f8un false https://oidc-r7u1.s3.us-east-1.amazonaws.com
aws:secretsmanager:us-east-1:242819244:secret:rosa-private-key-oidc-r7u1-tM3MDN
```

2.1.4. Creating Operator roles and policies

When using a ROSA with HCP cluster, you must create the Operator IAM roles that are required for Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) deployments. The cluster Operators use the Operator roles to obtain the temporary permissions required to carry out cluster operations, such as managing back-end storage, cloud provider credentials, and external access to a cluster.

Prerequisites

- You have completed the AWS prerequisites for ROSA with HCP.
- You have installed and configured the latest Red Hat OpenShift Service on AWS ROSA CLI (**rosa**), on your installation host.
- You created the account-wide AWS roles.

Procedure

1. Set your prefix name to an environment variable using the following command:

```
$ export OPERATOR_ROLES_PREFIX=<prefix_name>
```

2. To create your Operator roles, run the following command:

```
$ rosa create operator-roles --hosted-cp --prefix=$OPERATOR_ROLES_PREFIX --oidc-
config-id=$OIDC_ID --installer-role-arn
arn:aws:iam::${AWS_ACCOUNT_ID}:role/${ACCOUNT_ROLES_PREFIX}-HCP-ROSA-
Installer-Role
```

The following breakdown provides options for the Operator role creation.

```
$ rosa create operator-roles --hosted-cp
--prefix=$OPERATOR_ROLES_PREFIX 1
--oidc-config-id=$OIDC_ID 2
--installer-role-arn
arn:aws:iam::${AWS_ACCOUNT_ID}:role/${ACCOUNT_ROLES_PREFIX}-HCP-ROSA-
Installer-Role 3
```

- 1** You must supply a prefix when creating these Operator roles. Failing to do so produces an error. See the Additional resources of this section for information on the Operator prefix.
- 2** This value is the OIDC configuration ID that you created for your ROSA with HCP cluster.
- 3** This value is the installer role ARN that you created when you created the ROSA account roles.

You must include the **--hosted-cp** parameter to create the correct roles for ROSA with HCP clusters. This command returns the following information.

Sample output

```
? Role creation mode: auto
? Operator roles prefix: <pre-filled_prefix> 1
? OIDC Configuration ID: 23soa2bgvpek9kmes9s7os0a39i13qm4 |
https://dvbwdgztaeq9o.cloudfront.net/23soa2bgvpek9kmes9s7os0a39i13qm4 2
? Create hosted control plane operator roles: Yes
W: More than one Installer role found
? Installer role ARN: arn:aws:iam::4540112244:role/<prefix>-HCP-ROSA-Installer-Role
? Permissions boundary ARN (optional):
I: Reusable OIDC Configuration detected. Validating trusted relationships to operator roles:
I: Creating roles using 'arn:aws:iam::4540112244:user:<userName>'
I: Created role '<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials'
I: Created role '<prefix>-openshift-cloud-network-config-controller-cloud-credenti' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-cloud-network-config-controller-cloud-
credenti'
I: Created role '<prefix>-kube-system-kube-controller-manager' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-kube-controller-manager'
I: Created role '<prefix>-kube-system-capac-controller-manager' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-capac-controller-manager'
```

```
I: Created role '<prefix>-kube-system-control-plane-operator' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-control-plane-operator'
I: Created role '<prefix>-kube-system-kms-provider' with ARN
'arn:aws:iam::4540112244:role/<prefix>-kube-system-kms-provider'
I: Created role '<prefix>-openshift-image-registry-installer-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-image-registry-installer-cloud-credentials'
I: Created role '<prefix>-openshift-ingress-operator-cloud-credentials' with ARN
'arn:aws:iam::4540112244:role/<prefix>-openshift-ingress-operator-cloud-credentials'
I: To create a cluster with these roles, run the following command:
rosa create cluster --sts --oidc-config-id 23soa2bgvpek9kmes9s7os0a39i13qm4 --operator-
roles-prefix <prefix> --hosted-cp
```

- 1 This field is prepopulated with the prefix that you set in the initial creation command.
- 2 This field requires you to select an OIDC configuration that you created for your ROSA with HCP cluster.

The Operator roles are now created and ready to use for creating your ROSA with HCP cluster.

Verification

- You can list the Operator roles associated with your ROSA account. Run the following command:

```
$ rosa list operator-roles
```

Sample output

```
I: Fetching operator roles
ROLE PREFIX AMOUNT IN BUNDLE
<prefix> 8
? Would you like to detail a specific prefix Yes 1
? Operator Role Prefix: <prefix>
ROLE NAME ROLE ARN
VERSION MANAGED
<prefix>-kube-system-capac-controller-manager
arn:aws:iam::4540112244:role/<prefix>-kube-system-capac-controller-manager
4.13 No
<prefix>-kube-system-control-plane-operator
arn:aws:iam::4540112244:role/<prefix>-kube-system-control-plane-operator
4.13 No
<prefix>-kube-system-kms-provider
arn:aws:iam::4540112244:role/<prefix>-kube-system-kms-provider 4.13
No
<prefix>-kube-system-kube-controller-manager
arn:aws:iam::4540112244:role/<prefix>-kube-system-kube-controller-manager
4.13 No
<prefix>-openshift-cloud-network-config-controller-cloud-credenti
arn:aws:iam::4540112244:role/<prefix>-openshift-cloud-network-config-controller-cloud-
credenti 4.13 No
<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials
arn:aws:iam::4540112244:role/<prefix>-openshift-cluster-csi-drivers-ebs-cloud-credentials
4.13 No
<prefix>-openshift-image-registry-installer-cloud-credentials
```

```
arn:aws:iam::4540112244:role/<prefix>-openshift-image-registry-installer-cloud-credentials
4.13 No
<prefix>-openshift-ingress-operator-cloud-credentials
arn:aws:iam::4540112244:role/<prefix>-openshift-ingress-operator-cloud-credentials
4.13 No
```

- 1 After the command runs, it displays all the prefixes associated with your AWS account and notes how many roles are associated with this prefix. If you need to see all of these roles and their details, enter "Yes" on the detail prompt to have these roles listed out with specifics.

2.1.5. Creating a ROSA cluster using a custom AWS KMS key

You can create a Red Hat OpenShift Service on AWS (ROSA) cluster with a customer-provided KMS key that is used to encrypt either node root volumes, the etcd database, or both. A different KMS key ARN can be provided for each option.



NOTE

ROSA with HCP does not automatically configure the **default** storage class to encrypt persistent volumes with the customer-provided KMS key. This is something that can be configured in-cluster after installation.

Procedure

1. Create a custom AWS customer-managed KMS key by running the following command:

```
$ KMS_ARN=$(aws kms create-key --region $AWS_REGION --description 'Custom ROSA Encryption Key' --tags TagKey=red-hat,TagValue=true --query KeyMetadata.Arn --output text)
```

This command saves the Amazon Resource Name (ARN) output of this custom key for further steps.



NOTE

Customers must provide the **--tags TagKey=red-hat,TagValue=true** argument that is required for a customer KMS key.

2. Verify the KMS key has been created by running the following command:

```
$ echo $KMS_ARN
```

3. Set your AWS account ID to an environment variable.

```
$ AWS_ACCOUNT_ID=<aws_account_id>
```

4. Add the ARN for the account-wide installer role and operator roles that you created in the preceding step to the **Statement.Principal.AWS** section in the file. In the following example, the ARN for the default **ManagedOpenShift-HCP-ROSA-Installer-Role** role is added:

```
{
```

```

"Version": "2012-10-17",
"Id": "key-rosa-policy-1",
"Statement": [
{
  "Sid": "Enable IAM User Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::${AWS_ACCOUNT_ID}:root"
  },
  "Action": "kms:*",
  "Resource": "*"
},
{
  "Sid": "Installer Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::${AWS_ACCOUNT_ID}:role/ManagedOpenShift-HCP-
ROSA-Installer-Role"
  },
  "Action": [
    "kms:CreateGrant",
    "kms:DescribeKey",
    "kms:GenerateDataKeyWithoutPlaintext"
  ],
  "Resource": "*"
},
{
  "Sid": "ROSA KubeControllerManager Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::${AWS_ACCOUNT_ID}:role/<operator_role_prefix>-kube-
system-kube-controller-manager"
  },
  "Action": "kms:DescribeKey",
  "Resource": "*"
},
{
  "Sid": "ROSA KMS Provider Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::${AWS_ACCOUNT_ID}:role/<operator_role_prefix>-kube-
system-kms-provider"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:DescribeKey"
  ],
  "Resource": "*"
},
{
  "Sid": "ROSA NodeManager Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::${AWS_ACCOUNT_ID}:role/<operator_role_prefix>-kube-

```

```

system-capa-controller-manager"
    },
    "Action": [
        "kms:DescribeKey",
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:CreateGrant"
    ],
    "Resource": "*"
  }
]
}

```

- Confirm the details of the policy file created by running the following command:

```
$ cat rosa-key-policy.json
```

- Apply the newly generated key policy to the custom KMS key by running the following command:

```
$ aws kms put-key-policy --key-id $KMS_ARN \
--policy file://rosa-key-policy.json \
--policy-name default
```

- Create the cluster by running the following command:



NOTE

If your cluster name is longer than 15 characters, it will contain an autogenerated domain prefix as a sub-domain for your provisioned cluster on ***.openshiftapps.com**.

To customize the subdomain, use the **--domain-prefix** flag. The domain prefix cannot be longer than 15 characters, must be unique, and cannot be changed after cluster creation.

```
$ rosa create cluster --cluster-name <cluster_name> \
--subnet-ids <private_subnet_id>,<public_subnet_id> \
--sts \
--mode auto \
--machine-cidr 10.0.0.0/16 \
--compute-machine-type m5.xlarge \
--hosted-cp \
--region <aws_region> \
--oidc-config-id $OIDC_ID \
--kms-key-arn $KMS_ARN \ 1
--etcd-encryption-kms-arn $KMS_ARN \ 2
--operator-roles-prefix $OPERATOR_ROLES_PREFIX
```

- This KMS key ARN is used to encrypt all worker node root volumes. It is not required if only etcd database encryption is needed.
- This KMS key ARN is used to encrypt the etcd database. The etcd database is always encrypted by default with an AES cipher block, but can be encrypted instead with a KMS key. It is not required if only node root volume encryption is needed.

Verification

You can verify that your KMS key works by using [OpenShift Cluster Manager](#).

1. Navigate to [OpenShift Cluster Manager](#) and select **Instances**.
2. Select your instance.
3. Click the **Storage** tab.
4. Copy the **KMS key ID**.
5. Search and select **Key Management Service**.
6. Enter your copied *KMS key ID* in the **Filter** field.

2.2. NEXT STEPS

- [Accessing a ROSA cluster](#)

2.3. ADDITIONAL RESOURCES

- For information on using the CLI to create a cluster, see [Creating a ROSA with HCP cluster using the CLI](#).
- For steps to deploy a ROSA cluster using manual mode, see [Creating a cluster using customizations](#).
- For more information about the AWS Identity Access Management (IAM) resources required to deploy Red Hat OpenShift Service on AWS with STS, see [About IAM resources for clusters that use STS](#).
- For details about optionally setting an Operator role name prefix, see [About custom Operator IAM role prefixes](#).
- For information about the prerequisites to installing ROSA with STS, see [AWS prerequisites for ROSA with STS](#).
- For details about using the **auto** and **manual** modes to create the required STS resources, see [Understanding the auto and manual deployment modes](#).
- For more information about using OpenID Connect (OIDC) identity providers in AWS IAM, see [Creating OpenID Connect \(OIDC\) identity providers](#).
- For more information about troubleshooting ROSA cluster installations, see [Troubleshooting installations](#).
- For steps to contact Red Hat Support for assistance, see [Getting support for Red Hat OpenShift Service on AWS](#).

CHAPTER 3. CREATING A PRIVATE CLUSTER ON ROSA WITH HCP

This document describes how to create a Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) private cluster.

3.1. CREATING AN AWS PRIVATE CLUSTER

You can create a private cluster with multiple availability zones (Multi-AZ) on ROSA with HCP using the ROSA command line interface (CLI), **rosa**.

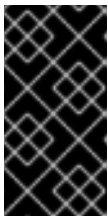
Prerequisites

- You have available AWS service quotas.
- You have enabled the ROSA service in the AWS Console.
- You have installed and configured the latest version of the ROSA CLI on your installation host.

Procedure

Creating a cluster with hosted control planes can take around 10 minutes.

1. Create a VPC with at least one private subnet. Ensure that your machine's classless inter-domain routing (CIDR) matches your virtual private cloud's CIDR. For more information, see [Requirements for using your own VPC](#) and [VPC Validation](#).



IMPORTANT

If you use a firewall, you must configure it so that ROSA can access the sites that required to function.

For more information, see the "AWS PrivateLink firewall prerequisites" section.

2. Create the account-wide IAM roles by running the following command:

```
$ rosa create account-roles --hosted-cp
```

3. Create the OIDC configuration by running the following command:

```
$ rosa create oidc-config --mode=auto --yes
```

Save the OIDC configuration ID because you need it to create the Operator roles.

Example output

```
I: Setting up managed OIDC configuration
I: To create Operator Roles for this OIDC Configuration, run the following command and
remember to replace <user-defined> with a prefix of your choice:
  rosa create operator-roles --prefix <user-defined> --oidc-config-id
  28s4avcdt2l318r1jbk3ifmimkurk384
If you are going to create a Hosted Control Plane cluster please include '--hosted-cp'
```



```
I: Creating OIDC provider using 'arn:aws:iam::46545644412:user/user'
I: Created OIDC provider with ARN 'arn:aws:iam::46545644412:oidc-
provider/oidc.op1.openshiftapps.com/28s4avcdt2l318r1jbk3ifmimkurk384'
```

4. Create the Operator roles by running the following command:

```
$ rosa create operator-roles --hosted-cp --prefix <operator_roles_prefix> --oidc-config-id
<oidc_config_id> --installer-role-arn
arn:aws:iam::<account_roles_prefix>:role/<account_roles_prefix>-HCP-ROSA-Installer-
Role
```

5. Create a private ROSA with HCP cluster by running the following command:

```
$ rosa create cluster --private --cluster-name=<cluster-name> --sts --mode=auto --hosted-cp
--operator-roles-prefix <operator_role_prefix> --oidc-config-id <oidc_config_id> [--machine-
cidr=<VPC CIDR>/16] --subnet-ids=<private-subnet-id1>[,<private-subnet-id2>,<private-
subnet-id3>]
```

6. Enter the following command to check the status of your cluster. During cluster creation, the **State** field from the output will transition from **pending** to **installing**, and finally, to **ready**.

```
$ rosa describe cluster --cluster=<cluster_name>
```



NOTE

If installation fails or the **State** field does not change to **ready** after 10 minutes, see the "Troubleshooting Red Hat OpenShift Service on AWS installations" documentation in the Additional resources section.

7. Enter the following command to follow the OpenShift installer logs to track the progress of your cluster:

```
$ rosa logs install --cluster=<cluster_name> --watch
```

3.2. CONFIGURING AWS SECURITY GROUPS TO ACCESS THE API

With ROSA with HCP private clusters, the AWS PrivateLink endpoint exposed in the customer's VPC has a default security group. This security group has access to the PrivateLink endpoint that is limited to only those resources that exist within the VPC or resources that are present with an IP address associated with the VPC CIDR range. In order to grant access to any entities outside of the VPC, through VPC peering and transit gateway, you must create and attach another security group to the PrivateLink endpoint to grant the necessary access.

Prerequisites

- Your corporate network or other VPC has connectivity.
- You have permission to create and attach security groups within the VPC.

Procedure

1. Set your cluster name as an environmental variable by running the following command:

-

```
$ export CLUSTER_NAME=<cluster_name>
```

You can verify that the variable has been set by running the following command:

```
$ echo $CLUSTER_NAME
```

Example output

```
hcp-private
```

- Find the VPC endpoint (VPCE) ID and VPC ID by running the following command:

```
$ read -r VPCE_ID VPC_ID <<< $(aws ec2 describe-vpc-endpoints --filters
"Name=tag:api.openshift.com/id,Values=$(rosa describe cluster -c ${CLUSTER_NAME} -o
yaml | grep '^id: ' | cut -d' ' -f2)" --query 'VpcEndpoints[][VpcEndpointId,VpcId]' --output text)
```

- Create your security group by running the following command:

```
$ export SG_ID=$(aws ec2 create-security-group --description "Granting API access to
${CLUSTER_NAME} from outside of VPC" --group-name "${CLUSTER_NAME}-api-sg" --
vpc-id $VPC_ID --output text)
```

- Add an ingress rule to the security group by running the following command:

```
$ aws ec2 authorize-security-group-ingress --group-id $SG_ID --ip-permissions
FromPort=443,ToPort=443,IpProtocol=tcp,IpRanges=[{CidrIp=0.0.0.0/0}]
```

- Add the new security group to the VPCE by running the following command:

```
$ aws ec2 modify-vpc-endpoint --vpc-endpoint-id $VPCE_ID --add-security-group-ids
$SG_ID
```

You now can access the API with your ROSA with HCP private cluster.

3.3. NEXT STEPS

[Configuring identity providers](#)

3.4. ADDITIONAL RESOURCES

- [AWS PrivateLink firewall prerequisites](#)
- [Overview of the ROSA with STS deployment workflow](#)
- [Deleting a ROSA cluster](#)
- [ROSA architecture models](#)
- [Troubleshooting Red Hat OpenShift Service on AWS installations](#)

CHAPTER 4. USING THE NODE TUNING OPERATOR ON ROSA WITH HCP CLUSTERS

Red Hat OpenShift Service on AWS (ROSA) with hosted control planes (HCP) supports the Node Tuning Operator to improve performance of your nodes on your ROSA with HCP clusters. Prior to creating a node tuning configuration, you must create a custom tuning specification.

Purpose

The Node Tuning Operator helps you manage node-level tuning by orchestrating the TuneD daemon and achieves low latency performance by using the Performance Profile controller. The majority of high-performance applications require some level of kernel tuning. The Node Tuning Operator provides a unified management interface to users of node-level sysctls and more flexibility to add custom tuning specified by user needs.

The Operator manages the containerized TuneD daemon for Red Hat OpenShift Service on AWS as a Kubernetes daemon set. It ensures the custom tuning specification is passed to all containerized TuneD daemons running in the cluster in the format that the daemons understand. The daemons run on all nodes in the cluster, one per node.

Node-level settings applied by the containerized TuneD daemon are rolled back on an event that triggers a profile change or when the containerized TuneD daemon is terminated gracefully by receiving and handling a termination signal.

The Node Tuning Operator uses the Performance Profile controller to implement automatic tuning to achieve low latency performance for Red Hat OpenShift Service on AWS applications.

The cluster administrator configures a performance profile to define node-level settings such as the following:

- Updating the kernel to kernel-rt.
- Choosing CPUs for housekeeping.
- Choosing CPUs for running workloads.



NOTE

Currently, disabling CPU load balancing is not supported by cgroup v2. As a result, you might not get the desired behavior from performance profiles if you have cgroup v2 enabled. Enabling cgroup v2 is not recommended if you are using performance profiles.

The Node Tuning Operator is part of a standard Red Hat OpenShift Service on AWS installation in version 4.1 and later.



NOTE

In earlier versions of Red Hat OpenShift Service on AWS, the Performance Addon Operator was used to implement automatic tuning to achieve low latency performance for OpenShift applications. In Red Hat OpenShift Service on AWS 4.11 and later, this functionality is part of the Node Tuning Operator.

4.1. CUSTOM TUNING SPECIFICATION

The custom resource (CR) for the Operator has two major sections. The first section, **profile:**, is a list of Tuned profiles and their names. The second, **recommend:**, defines the profile selection logic.

Multiple custom tuning specifications can co-exist as multiple CRs in the Operator's namespace. The existence of new CRs or the deletion of old CRs is detected by the Operator. All existing custom tuning specifications are merged and appropriate objects for the containerized Tuned daemons are updated.

Management state

The Operator Management state is set by adjusting the default Tuned CR. By default, the Operator is in the Managed state and the **spec.managementState** field is not present in the default Tuned CR. Valid values for the Operator Management state are as follows:

- Managed: the Operator will update its operands as configuration resources are updated
- Unmanaged: the Operator will ignore changes to the configuration resources
- Removed: the Operator will remove its operands and resources the Operator provisioned

Profile data

The **profile:** section lists Tuned profiles and their names.

```
{
  "profile": [
    {
      "name": "tuned_profile_1",
      "data": "# Tuned profile specification\n[main]\nsummary=Description of tuned_profile_1\nprofile\n\n[sysctl]\nnet.ipv4.ip_forward=1\n# ... other sysctl's or other Tuned daemon plugins\nsupported by the containerized Tuned\n"
    },
    {
      "name": "tuned_profile_n",
      "data": "# Tuned profile specification\n[main]\nsummary=Description of tuned_profile_n\nprofile\n\n# tuned_profile_n profile settings\n"
    }
  ]
}
```

Recommended profiles

The **profile:** selection logic is defined by the **recommend:** section of the CR. The **recommend:** section is a list of items to recommend the profiles based on a selection criteria.

```
"recommend": [
  {
    "recommend-item-1": details_of_recommendation,
    # ...
    "recommend-item-n": details_of_recommendation,
  }
]
```

The individual items of the list:

```
{
  "profile": [
```

```

{
  # ...
}
],
"recommend": [
  {
    "profile": <tuned_profile_name>, ❶
    "priority": { <priority>, ❷
  },
  "match": [ ❸
    {
      "label": <label_information> ❹
    },
  ]
},
]
}

```

- ❶ A TuneD profile to apply on a match. For example **tuned_profile_1**.
- ❷ Profile ordering priority. Lower numbers mean higher priority (**0** is the highest priority).
- ❸ If omitted, profile match is assumed unless a profile with a higher priority matches first.
- ❹ The label for the profile matched items.

<match> is an optional list recursively defined as follows:

```

"match": [
  {
    "label": ❶
  },
]

```

- ❶ Node or pod label name.

If **<match>** is not omitted, all nested **<match>** sections must also evaluate to **true**. Otherwise, **false** is assumed and the profile with the respective **<match>** section will not be applied or recommended. Therefore, the nesting (child **<match>** sections) works as logical AND operator. Conversely, if any item of the **<match>** list matches, the entire **<match>** list evaluates to **true**. Therefore, the list acts as logical OR operator.

Example: Node or pod label based matching

```

[
  {
    "match": [
      {
        "label": "tuned.openshift.io/elasticsearch",
        "match": [
          {
            "label": "node-role.kubernetes.io/master"
          },
        ]
      },
    ]
  },
]

```

```

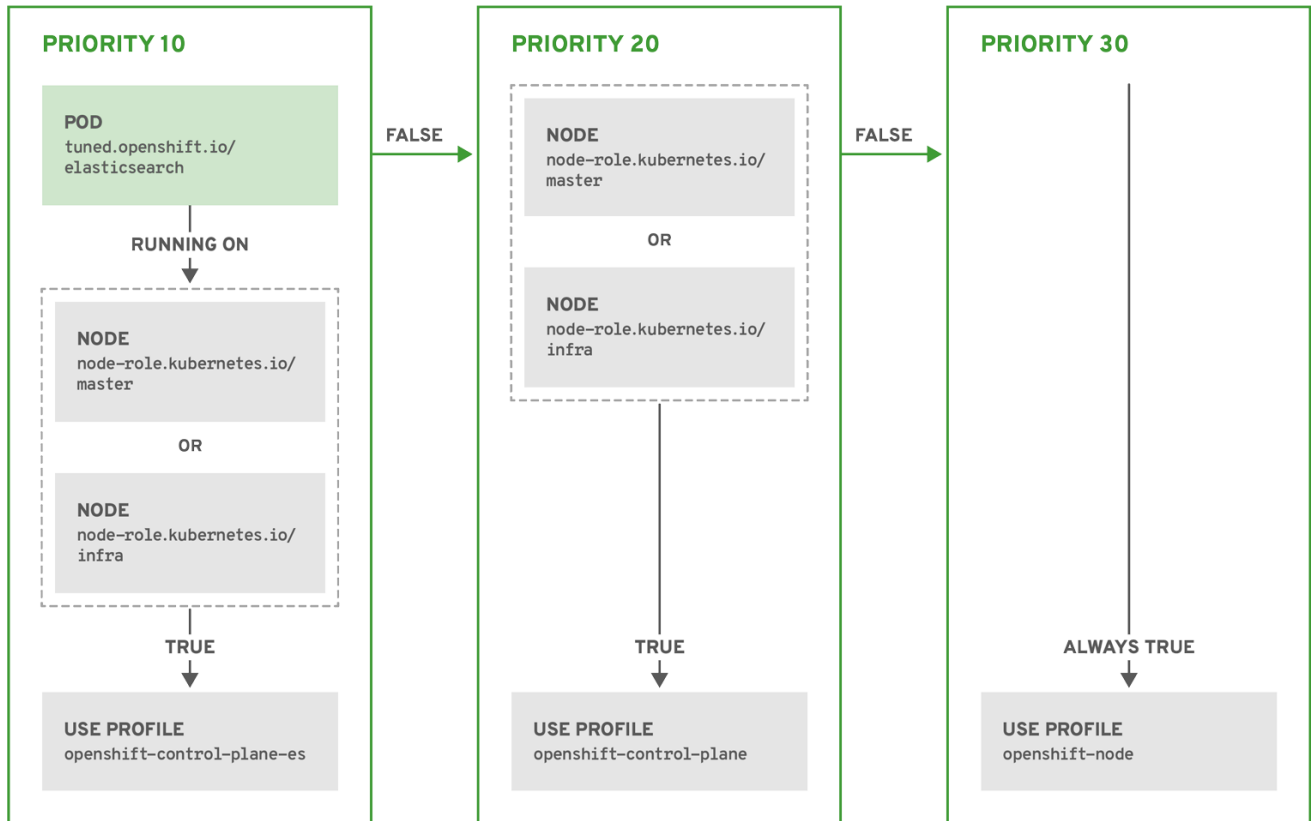
    {
      "label": "node-role.kubernetes.io/infra"
    }
  ],
  "type": "pod"
}
],
"priority": 10,
"profile": "openshift-control-plane-es"
},
{
  "match": [
    {
      "label": "node-role.kubernetes.io/master"
    },
    {
      "label": "node-role.kubernetes.io/infra"
    }
  ],
  "priority": 20,
  "profile": "openshift-control-plane"
},
{
  "priority": 30,
  "profile": "openshift-node"
}
]

```

The CR above is translated for the containerized TuneD daemon into its **recommend.conf** file based on the profile priorities. The profile with the highest priority (**10**) is **openshift-control-plane-es** and, therefore, it is considered first. The containerized TuneD daemon running on a given node looks to see if there is a pod running on the same node with the **tuned.openshift.io/elasticsearch** label set. If not, the entire **<match>** section evaluates as **false**. If there is such a pod with the label, in order for the **<match>** section to evaluate to **true**, the node label also needs to be **node-role.kubernetes.io/master** or **node-role.kubernetes.io/infra**.

If the labels for the profile with priority **10** matched, **openshift-control-plane-es** profile is applied and no other profile is considered. If the node/pod label combination did not match, the second highest priority profile (**openshift-control-plane**) is considered. This profile is applied if the containerized TuneD pod runs on a node with labels **node-role.kubernetes.io/master** or **node-role.kubernetes.io/infra**.

Finally, the profile **openshift-node** has the lowest priority of **30**. It lacks the **<match>** section and, therefore, will always match. It acts as a profile catch-all to set **openshift-node** profile, if no other profile with higher priority matches on a given node.



OPENSIFT_10_0319

Example: Machine pool based matching

```

{
  "apiVersion": "tuned.openshift.io/v1",
  "kind": "Tuned",
  "metadata": {
    "name": "openshift-node-custom",
    "namespace": "openshift-cluster-node-tuning-operator"
  },
  "spec": {
    "profile": [
      {
        "data": "[main]\nsummary=Custom OpenShift node profile with an additional kernel\nparameter\ninclude=openshift-node\n[bootloader]\ncmdline_openshift_node_custom=+skew_tick=1\n",
        "name": "openshift-node-custom"
      }
    ],
    "recommend": [
      {
        "priority": 20,
        "profile": "openshift-node-custom"
      }
    ]
  }
}

```

Cloud provider-specific Tuned profiles

With this functionality, all Cloud provider-specific nodes can conveniently be assigned a TuneD profile specifically tailored to a given Cloud provider on a Red Hat OpenShift Service on AWS cluster. This can be accomplished without adding additional node labels or grouping nodes into machine pools.

This functionality takes advantage of **spec.providerID** node object values in the form of **<cloud-provider>://<cloud-provider-specific-id>** and writes the file **/var/lib/tuned/provider** with the value **<cloud-provider>** in NTO operand containers. The content of this file is then used by TuneD to load **provider-<cloud-provider>** profile if such profile exists.

The **openshift** profile that both **openshift-control-plane** and **openshift-node** profiles inherit settings from is now updated to use this functionality through the use of conditional profile loading. Neither NTO nor TuneD currently include any Cloud provider-specific profiles. However, it is possible to create a custom profile **provider-<cloud-provider>** that will be applied to all Cloud provider-specific cluster nodes.

Example GCE Cloud provider profile

```
{
  "apiVersion": "tuned.openshift.io/v1",
  "kind": "Tuned",
  "metadata": {
    "name": "provider-gce",
    "namespace": "openshift-cluster-node-tuning-operator"
  },
  "spec": {
    "profile": [
      {
        "data": "[main]\nsummary=GCE Cloud provider-specific profile\n# Your tuning for GCE Cloud
provider goes here.\n",
        "name": "provider-gce"
      }
    ]
  }
}
```



NOTE

Due to profile inheritance, any setting specified in the **provider-<cloud-provider>** profile will be overwritten by the **openshift** profile and its child profiles.

4.2. CREATING NODE TUNING CONFIGURATIONS ON ROSA WITH HCP

You can create tuning configurations using the Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**.

Prerequisites

- You have downloaded the latest version of the ROSA CLI.
- You have a cluster on the latest version.
- You have a specification file configured for node tuning.

Procedure

1. Run the following command to create your tuning configuration:

```
$ rosa create tuning-config -c <cluster_id> --name <name_of_tuning> --spec-path
<path_to_spec_file>
```

You must supply the path to the **spec.json** file or the command returns an error.

Sample output

```
$ I: Tuning config 'sample-tuning' has been created on cluster 'cluster-example'.
$ I: To view all tuning configs, run 'rosa list tuning-configs -c cluster-example'
```

Validation

- You can verify the existing tuning configurations that are applied by your account with the following command:

```
$ rosa list tuning-configs -c <cluster_name> [-o json]
```

You can specify the type of output you want for the configuration list.

- Without specifying the output type, you see the ID and name of the tuning configuration:

Sample output without specifying output type

```
ID                NAME
20468b8e-edc7-11ed-b0e4-0a580a800298  sample-tuning
```

- If you specify an output type, such as **json**, you receive the tuning configuration as JSON text:



NOTE

The following JSON output has hard line-returns for the sake of reading clarity. This JSON output is invalid unless you remove the newlines in the JSON strings.

Sample output specifying JSON output

```
[
  {
    "kind": "TuningConfig",
    "id": "20468b8e-edc7-11ed-b0e4-0a580a800298",
    "href":
"/api/clusters_mgmt/v1/clusters/23jbsevqb22l0m58ps39ua4trff9179e/tuning_configs/20468b8e-edc7-11ed-b0e4-0a580a800298",
    "name": "sample-tuning",
    "spec": {
      "profile": [
        {
          "data": "[main]\nsummary=Custom OpenShift profile\ninclude=openshift-node\n\n[sysctl]nvm.dirty_ratio=55\n",
          "name": "tuned-1-profile"
        }
      ]
    }
  }
]
```

```

    }
  ],
  "recommend": [
    {
      "priority": 20,
      "profile": "tuned-1-profile"
    }
  ]
}
]

```

4.3. MODIFYING YOUR NODE TUNING CONFIGURATIONS FOR ROSA WITH HCP

You can view and update the node tuning configurations using the Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**.

Prerequisites

- You have downloaded the latest version of the ROSA CLI.
- You have a cluster on the latest version
- Your cluster has a node tuning configuration added to it

Procedure

1. You view the tuning configurations with the **rosa describe** command:

```

$ rosa describe tuning-config -c <cluster_id> ❶
  --name <name_of_tuning> ❷
  [-o json] ❸

```

The following items in this spec file are:

- ❶ Provide the cluster ID for the cluster that you own that you want to apply a node tuning configurations.
- ❷ Provide the name of your tuning configuration.
- ❸ Optionally, you can provide the output type. If you do not specify any outputs, you only see the ID and name of the tuning configuration.

Sample output without specifying output type

```

Name: sample-tuning
ID: 20468b8e-edc7-11ed-b0e4-0a580a800298
Spec: {
  "profile": [
    {
      "data": "[main]\nsummary=Custom OpenShift profile\ninclude=openshift-
node\n\n[sysctl]\nvmm.dirty_ratio=\"55\"\n",

```

```

        "name": "tuned-1-profile"
      }
    ],
    "recommend": [
      {
        "priority": 20,
        "profile": "tuned-1-profile"
      }
    ]
  }
}

```

Sample output specifying JSON output

```

{
  "kind": "TuningConfig",
  "id": "20468b8e-edc7-11ed-b0e4-0a580a800298",
  "href":
"/api/clusters_mgmt/v1/clusters/23jbsevqb2210m58ps39ua4trff9179e/tuning_configs/20468b8e-
edc7-11ed-b0e4-0a580a800298",
  "name": "sample-tuning",
  "spec": {
    "profile": [
      {
        "data": "[main]\nsummary=Custom OpenShift profile\ninclude=openshift-
node\n\n[sysctl]\nvm.dirty_ratio=\"55\"\n",
        "name": "tuned-1-profile"
      }
    ],
    "recommend": [
      {
        "priority": 20,
        "profile": "tuned-1-profile"
      }
    ]
  }
}

```

2. After verifying the tuning configuration, you edit the existing configurations with the **rosa edit** command:

```

$ rosa edit tuning-config -c <cluster_id> --name <name_of_tuning> --spec-path
<path_to_spec_file>

```

In this command, you use the **spec.json** file to edit your configurations.

Verification

- Run the **rosa describe** command again, to see that the changes you made to the **spec.json** file are updated in the tuning configurations:

```

$ rosa describe tuning-config -c <cluster_id> --name <name_of_tuning>

```

Sample output

```

Name: sample-tuning
ID: 20468b8e-edc7-11ed-b0e4-0a580a800298
Spec: {
  "profile": [
    {
      "data": "[main]\nsummary=Custom OpenShift profile\ninclude=openshift-
node\n\n[sysctl]\nvm.dirty_ratio=55\n",
      "name": "tuned-2-profile"
    }
  ],
  "recommend": [
    {
      "priority": 10,
      "profile": "tuned-2-profile"
    }
  ]
}

```

4.4. DELETING NODE TUNING CONFIGURATIONS ON ROSA WITH HCP

You can delete tuning configurations by using the Red Hat OpenShift Service on AWS (ROSA) CLI, **rosa**.



NOTE

You cannot delete a tuning configuration referenced in a machine pool. You must first remove the tuning configuration from all machine pools before you can delete it.

Prerequisites

- You have downloaded the latest version of the ROSA CLI.
- You have a cluster on the latest version .
- Your cluster has a node tuning configuration that you want delete.

Procedure

- To delete the tuning configurations, run the following command:

```
$ rosa delete tuning-config -c <cluster_id> <name_of_tuning>
```

The tuning configuration on the cluster is deleted

Sample output

```
? Are you sure you want to delete tuning config sample-tuning on cluster sample-cluster? Yes
! Successfully deleted tuning config 'sample-tuning' from cluster 'sample-cluster'
```

