Red Hat Advanced Cluster Management for Kubernetes 2.2

Manage cluster

Read more to learn how to create, import, and manage clusters across cloud providers.

Last Updated: 2022-05-18

Red Hat Advanced Cluster Management for Kubernetes 2.2 Manage cluster

Read more to learn how to create, import, and manage clusters across cloud providers.

Abstract

Read more to learn how to create, import, and manage clusters across cloud providers.

Table of Contents

CHAPTER 1. MANAGING YOUR CLUSTERS

Learn how to create, import, and manage clusters across cloud providers by using both the Red Hat Advanced Cluster Management for Kubernetes console.

Learn how to manage clusters across cloud providers in the following topics:

- Supported clouds
- Resizing a cluster
- Creating a provider connection
- Creating a cluster
- Importing a target managed cluster to the hub cluster
- ManagedClusterSets
- Creating an AnsibleJob for a managed cluster
- Submariner
- Upgrading your cluster
- Removing a cluster from management

CHAPTER 2. SUPPORTED CLOUDS

Learn about the cloud providers that are available with Red Hat Advanced Cluster Management for Kubernetes. Also, find the documented managed providers that are available.

- Supported hub cluster providers
- Supported managed cluster providers
- Configuring kubectl

Best practice: For managed cluster providers, use the latest version of Kubernetes.

2.1. SUPPORTED HUB CLUSTER PROVIDERS

Red Hat OpenShift Container Platform 4.5.2 or later, 4.6.1 or later, and 4.7.0 or later, are supported for the hub cluster.

- See OpenShift on Amazon Web Services.
- See Azure Red Hat OpenShift.
- See Red Hat OpenShift Dedicated.
- See Red Hat OpenShift Container Platform on OpenStack (OpenStack version 16.1, or later).
- See Red Hat OpenShift Container Platform on VMware vSphere .
- See Red Hat OpenShift Container Platform on IBM Cloud (ROKS) (Red Hat OpenShift Container Platform version 4.5, and later).

2.2. SUPPORTED MANAGED CLUSTER PROVIDERS

Red Hat OpenShift Container Platform 3.11.200 or later, 4.4.3 or later, 4.5.2 or later, 4.6.1 or later, and 4.7.0 or later, are supported for the managed clusters.

See the available managed cluster options and documentation:

- See OpenShift on Amazon Web Services.
- See Red Hat OpenShift Container Platform on IBM Cloud (ROKS) (Kubernetes 1.17, and later).
- See About Red Hat OpenShift Kubernetes Engine .
- See Red Hat OpenShift Container Platform 4.6.1, and later, on IBM Z .
- See Getting started with IBM Cloud Kubernetes Service (Kubernetes 1.18, and later).
- See Google Kubernetes Engine (Kubernetes 1.17, and later).
- See Azure Kubernetes Service (Kubernetes 1.19.6, and later).
- See Amazon Elastic Kubernetes Service (Kubernetes 1.17.6, and later).
- See Red Hat OpenShift Container Platform on VMware vSphere .

- See Azure Red Hat OpenShift.
- See Red Hat OpenShift Dedicated (Red Hat OpenShift Container Platform version 4.5.16, and later).
- See Red Hat OpenShift Container Platform on OpenStack (OpenStack version 16.1, or later).

2.3. CONFIGURING KUBECTL

From vendor documentation previously listed, you might need to learn how configure your **kubectl**. You must have **kubectl** installed when you import a managed cluster to a hub cluster. See Importing a target managed cluster to the hub cluster for details.

CHAPTER 3. RESIZING A CLUSTER

You can customize your Red Hat OpenShift Container Platform managed cluster specifications by using the scaling procedures, such as virtual machine sizes and number of nodes. See Recommended cluster scaling practices for the OpenShift Container Platform version of your cluster for information about resizing your cluster.

Tip: If you created the cluster by using the Red Hat Advanced Cluster Management for Kubernetes console, then it is an OpenShift Container Platform cluster.

CHAPTER 4. RELEASE IMAGES

When you create a cluster on a provider by using Red Hat Advanced Cluster Management for Kubernetes, you must specify a release image to use for the new cluster. The release image specifies which version of Red Hat OpenShift Container Platform is used to build the cluster.

The files that reference the release images are **yaml** files that are maintained in the **acm-hive-openshift-releases** GitHub repository. Red Hat Advanced Cluster Management uses those files to create the list of the available release images in the console. This includes the latest fast channel images from OpenShift Container Platform. The console only displays the latest release images for the three latest versions of OpenShift Container Platform. For example, you might see the following release images displayed in the console options:

- quay.io/openshift-release-dev/ocp-release:4.5.36-x86_64
- quay.io/openshift-release-dev/ocp-release:4.6.23-x86_64
- quay.io/openshift-release-dev/ocp-release:4.7.4-x86_64

Additional release images are stored, but are not visible in the console. To view all of the available release images, run **kubectl get clusterimageset** in your CLI. Only the latest versions are in the console to encourage the creation of clusters with the latest release images. In some cases, you might need to create a cluster that is a specific version, which is why the older versions are available.

The repository contains the **clusterImageSets** directory and the **subscription** directory, which are the directories that you use when working with the release images.

The **clusterImageSets** directory contains the following directories:

- Fast - Contains files that reference the latest versions of the release images for each OpenShift Container Platform version that is supported. The release images in this folder are tested, verified and supported.
- Releases - Contains files that reference all of the release images for each OpenShift Container Platform version (stable, fast, and candidate channels) **Note:** These releases have not all been tested and determined to be stable.
- Stable - Contains files that reference the latest two stable versions of the release images for each OpenShift Container Platform version that is supported. The release images in this folder are tested, verified and supported.

You can curate your own **ClusterImageSets** in three ways:

The first step for any of the three ways is to disable the included subscription that performs the automatic update of the latest fast channel images. The automatic curation of the latest fast **ClusterImageSets** can be disabled by using an installer parameter on the multiclusterhub resource. By toggling the **spec.disableUpdateClusterImageSets** parameter between **true** and **false**, the subscription installed with Red Hat Advanced Cluster Management is disabled or enabled, respectively. If you want to curate your own images, set the **spec.disableUpdateClusterImageSets** to **true** to disable the subscription.

Option 1: Specify the image reference for the specific **ClusterImageSet** that you want to use in the console when creating a cluster. Each new entry you specify persists and is available for all future cluster provisions. An example of an entry is: **quay.io/openshift-release-dev/ocp-release:4.6.8-x86_64**.

OPTION 2: Manually create and apply a **ClusterImageSets** YAML file from the **acm-hive-openshift-releases** GitHub repository.

OPTION 3: Follow the **README.md** in the **acm-hive-openshift-releases** GitHub repository to enable automatic updates of **ClusterImageSets** from a forked GitHub repository.

The **subscription** directory contains files that specify where the list of release images is pulled from. The default release images for Red Hat Advanced Cluster Management are provided in a Quay.io directory. The images are referenced by the files in the acm-hive-openshift-releases GitHub repository.

4.1. SYNCHRONIZING AVAILABLE RELEASE IMAGES

The release images are updated frequently, so you might want to synchronize the list of release images to ensure that you can select the latest available versions. The release images are available in the acm-hive-openshift-releases GitHub repository.

There are three levels of stability of the release images:

Table 4.1. Stability levels of release images

Category	Description
stable	Fully tested images that are confirmed to install and build clusters correctly.
fast	Partially tested, but likely less stable than a stable version.
candidate	Not tested, but the most current image. Might have some bugs.

Complete the following steps to refresh the list:

1. If the installer-managed **acm-hive-openshift-releases** subscription is enabled, disable the subscription by setting the value of **disableUpdateClusterImageSets** to **true**. You can remove the subscription by entering a command that is similar to the following command:

   ```
   oc delete -f subscription/subscription-stable
   ```

2. Clone the acm-hive-openshift-releases GitHub repository.

3. Connect to the stable release images and synchronize your Red Hat Advanced Cluster Management for Kubernetes hub cluster by entering the following command:

   ```
   make subscribe-stable
   ```

   Note: You can only run this **make** command when you are using the Linux or MacOS operating system.

   After about one minute, the latest list of **stable** release images is available.

   - To synchronize and display the fast release images, enter the following command:

     ```
     make subscribe-fast
     ```

Note: You can only run this **make** command when you are using the Linux or MacOS operating system.

About one minute after running the command, the list of available **stable** and **fast** release images updates with the currently available images.

- To synchronize and display the **candidate** release images, enter the following command:

  ```
  make subscribe-candidate
  ```

  Note: You can only run this **make** command when you are using the Linux or MacOS operating system.

  About one minute after running the command, the list of available **stable**, **fast**, and **candidate** release images updates with the currently available images.

4. View the list of currently available release images in the Red Hat Advanced Cluster Management console when you are creating a cluster.

5. You can unsubscribe from any of these channels to stop viewing the updates by entering a command in the following format:

   ```
   oc delete -f subscription/subscription-stable
   ```

4.1.1. Maintaining a custom list of release images when connected

You might want to ensure that you use the same release image for all of your clusters. To simplify, you can create your own custom list of release images that are available when creating a cluster. Complete the following steps to manage your available release images:

1. If the installer-managed **acm-hive-openshift-releases** subscription is enabled, disable it by setting the value of **disableUpdateClusterImageSets** to **true**.

2. Fork the acm-hive-openshift-releases GitHub repository.

3. Update the **./subscription/channel.yaml** file by changing the **spec: pathname** to access your the GitHub name for your forked repository, instead of **stolostron**. This step specifies where the hub cluster retrieves the release images. Your updated content should look similar to the following example:

   ```
   spec:
     type: GitHub
     pathname: https://github.com/<forked_content>/acm-hive-openshift-releases.git
   ```

   Replace *forked_content* with the path to your forked repository.

4. Add the YAML files for the images that you want available when you create a cluster by using the Red Hat Advanced Cluster Management for Kubernetes console to the *./clusterImageSets/stable/* ***or ./clusterImageSets/fast/* directory. *Tip:*** **You can retrieve the available YAML files from the main repository by merging changes into your forked repository.**

5. **Commit and merge your changes to your forked repository.**

6. **To synchronize your list of stable release images after you have cloned the acm-hive-openshift-releases repository, enter the following command to update the stable images:**

```
make subscribe-stable
```

Note: You can only run this**make** command when you are using the Linux or MacOS operating system.

After running this command, the list of available stable release images updates with the currently available images in about one minute.

7. By default, only the stable images are listed. To synchronize and display the fast release images, enter the following command:

```
make subscribe-fast
```

Note: You can only run this**make** command when you are using the Linux or MacOS operating system.

After running this command, the list of available fast release images updates with the currently available images in about 1 minute.

8. By default, Red Hat Advanced Cluster Management pre-loads a few ClusterImageSets. You can use the following commands to list what is available and remove the defaults.

```
oc get clusterImageSets
oc delete clusterImageSet <clusterImageSet_NAME>
```

Note: If you have not disabled the installer-managed automatic updates of the **ClusterImageSets** by setting the value of**disableUpdateClusterImageSets** to **true**, any images that you delete are recreated automatically.

9. View the list of currently available release images in the Red Hat Advanced Cluster Management console when you are creating a cluster.

4.1.2. Maintaining a custom list of release images while disconnected

In some cases, you need to maintain a custom list of release images when the hub cluster has no Internet connection. You can create your own custom list of release images that are available when creating a cluster. Complete the following steps to manage your available release images while disconnected:

1. While you are on a connected system, navigate to the acm-hive-openshift-releases GitHub repository.

2. Copy the **clusterImageSets** directory to a system that can access the disconnected Red Hat Advanced Cluster Management for Kubernetes hub cluster.

3. Add the YAML files for the images that you want available when you create a cluster by using the Red Hat Advanced Cluster Management console by manually adding the **clusterImageSet** YAML content.

4. Create **clusterImageSets** command:

```
oc create -f <clusterImageSet_FILE>
```

After running this command for each resource you want to add, the list of available release images will be available.

5. Alternately you can paste the image URL directly in the the create cluster console in Red Hat Advanced Cluster Management. This will create new clusterImageSets if they do not exist.

6. View the list of currently available release images in the Red Hat Advanced Cluster Management console when you are creating a cluster.

CHAPTER 5. CREATING AND MODIFYING BARE METAL ASSETS

Bare metal assets are virtual or physical servers that are configured to run your cloud operations. Red Hat Advanced Cluster Management for Kubernetes connects to a bare metal asset that your administrator creates, and can create clusters on it.

You must create a bare metal asset in Red Hat Advanced Cluster Management for Kubernetes to create a cluster on it. Use the following procedure to create a bare metal asset that can host a cluster that is managed by Red Hat Advanced Cluster Management for Kubernetes.

5.1. PREREQUISITES

You need the following prerequisites before creating a bare metal asset:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster on OpenShift Container Platform version 4.5, or later.
- Access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster to connect to the bare metal asset.
- A configured bare metal asset, and log in credentials with the required permissions to log in and manage it. Note: Login credentials for your bare metal asset include the following items for the asset that are provided by your administrator:
  - user name
  - password
  - Baseboard Management Controller Address
  - boot NIC MAC address

5.2. CREATING A BARE METAL ASSET WITH THE CONSOLE

To create a bare metal asset using the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Bare metal assets.
2. On the *Bare metal assets* page, ClickCreate bare metal asset
3. Enter a name for your asset that identifies it when you create a cluster.
4. Enter the namespace where you want to create the bare metal asset.
   Note: The bare metal asset, managed bare metal cluster, and its related secret must be in the same namespace.

   Users who have access to this namespace can associate this asset to the cluster when creating a cluster.
5. Enter the Baseboard Management Controller address. This is the controller that enables communication with the host. The following protocols are supported:
   - IPMI, see IPMI 2.0 Specification for more information.

- iDRAC, see Support for Integrated Dell Remote Access Controller 9 (iDRAC9) for more information.
- iRMC, see Data Sheet: FUJITSU Software ServerView Suite integrated Remote Management Controller - iRMC S5 for more information.
- Redfish, see Redfish specification for more information.

6. Enter the username and password for the bare metal asset.
7. Add the boot NIC MAC address for the bare metal asset. This is the MAC address of the host's network-connected NIC that is used to provision the host on the bare metal asset.

You can continue with Creating a cluster on bare metal.

5.3. MODIFYING A BARE METAL ASSET

If you need to modify the settings for a bare metal asset, complete the following steps:

1. In the Red Hat Advanced Cluster Management for Kubernetes console navigation, select: Automate infrastructure > Bare metal assets.
2. Select the options menu for the asset that you want to modify in the table.
3. Select Edit asset.

5.4. REMOVING A BARE METAL ASSET

When a bare metal asset is no longer used for any of the clusters, you can remove it from the list of available bare metal assets. Removing unused assets both simplifies your list of available assets, and prevents the accidental selection of that asset.

To remove a bare metal asset, complete the following steps:

1. In the Red Hat Advanced Cluster Management for Kubernetes console navigation, select: Automate infrastructure > Bare metal assets.
2. Select the options menu for the asset that you want to remove in the table.
3. Select Delete asset.

CHAPTER 6. CREATING A PROVIDER CONNECTION

A *provider connection* is required to create a Red Hat OpenShift Container Platform cluster on a cloud service provider with Red Hat Advanced Cluster Management for Kubernetes.

The provider connection stores the access credentials and configuration information for a provider. Each provider account requires its own provider connection, as does each domain on a single provider.

The following files detail the information that is required for creating a connection document for each supported provider:

- Creating a provider connection for Amazon Web Services
- Creating a provider connection for Microsoft Azure
- Creating a provider connection for Google Cloud Platform
- Creating a provider connection for VMware vSphere
- Creating a provider connection for bare metal

6.1. CREATING A PROVIDER CONNECTION FOR AMAZON WEB SERVICES

You need a provider connection to use Red Hat Advanced Cluster Management for Kubernetes console to deploy and manage an OpenShift cluster on Amazon Web Services (AWS).

Note: This procedure must be done before you can create a cluster with Red Hat Advanced Cluster Management for Kubernetes.

6.1.1. Prerequisites

You must have the following prerequisites before creating a provider connection:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on Amazon Web Services
- Amazon Web Services (AWS) login credentials, which include access key ID and secret access key. See Understanding and getting your security credentials.
- Account permissions that allow installing clusters on AWS. See Configuring an AWS account for instructions on how to configure.

6.1.2. Creating a provider connection by using the console

To create a provider connection from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, select the*Provider connections* tab.
   Existing provider connections are displayed.

3. Select Add a connection.

4. Select Amazon Web Services as your provider.

5. Add a name for your provider connection.

6. Select a namespace for your provider connection from the list.
   Tip: Create a namespace specifically to host your provider connections, both for convenience and added security.

7. You can optionally add a *Base DNS domain* for your provider connection. If you add the base DNS domain to the provider connection, it is automatically populated in the correct field when you create a cluster with this provider connection.

8. Add your *AWS access key ID* for your Amazon Web Services account. Log in to AWS to find the ID.

9. Add your *AWS Secret Access Key*.

10. Enter your *Red Hat OpenShift pull secret*. You can download your pull secret from Pull secret.

11. Add your *SSH private key* and *SSH public key*, which allows you to connect to the cluster. You can use an existing key pair, or create a new one with key generation program. See Generating an SSH private key and adding it to the agent for more information about how to generate a key.

12. Click Create. When you create the provider connection, it is added to the list of provider connections.

You can create a cluster that uses this provider connection by completing the steps in Creating a cluster on Amazon Web Services.

6.1.3. Deleting your provider connection

When you are no longer managing a cluster that is using a provider connection, delete the provider connection to protect the information in the provider connection.

1. From the navigation menu, navigate to Automate infrastructure > Clusters.

2. Select Provider connections.

3. Select the options menu beside the provider connection that you want to delete.

4. Select Delete connection.

6.2. CREATING A PROVIDER CONNECTION FOR MICROSOFT AZURE

You need a provider connection to use Red Hat Advanced Cluster Management for Kubernetes console to create and manage a Red Hat OpenShift Container Platform cluster on Microsoft Azure.

Note: This procedure is a prerequisite for creating a cluster with Red Hat Advanced Cluster Management for Kubernetes.

6.2.1. Prerequisites

You must have the following prerequisites before creating a provider connection:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster.
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so that it can create the Kubernetes cluster on Azure.
- Azure login credentials, which include your Base Domain Resource Group and Azure Service Principal JSON. See azure.microsoft.com.
- Account permissions that allow installing clusters on Azure. See How to configure Cloud Services and Configuring an Azure accountfor more information.

6.2.2. Creating a provider connection by using the console

To create a provider connection from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, select the*Provider connections* tab.
   Existing provider connections are displayed.
3. Select Add a connection.
4. Select Microsoft Azure as your provider.
5. Add a name for your provider connection.
6. Select a namespace for your provider connection from the list.
   Tip: You can create a namespace specifically to host your provider connections, both for convenience and added security.
7. You can optionally add a *Base DNS domain* for your provider connection. If you add the base DNS domain to the provider connection, it is automatically populated in the correct field when you create a cluster with this provider connection.
8. Add your *Base domain resource group name* for your Azure account. This entry is the resource name that you created with your Azure account. You can find your Base Domain Resource Group Name by selecting Home >DNS Zones in the Azure interface. Your Base domain resource group name is in the *Resource Group* column of the entry that contains the Base DNS domain that applies to your account.
9. Add your *Client ID*. This value is generated as the**appId** property when you create a service principal with the following command:

   ```
   az ad sp create-for-rbac --role Contributor --name <service_principal>
   ```

   Replace *service_principal* with the name of your service principal.
10. Add your *Client Secret*. This value is generated as the**password** property when you create a service principal with the following command:

    ```
    az ad sp create-for-rbac --role Contributor --name <service_principal>
    ```

    Replace *service_principal* with the name of your service principal.

11. Add your *Subscription ID*. This value is the **id** property in the output of the following command:

    ```
    az account show
    ```

12. Add your *Tenant ID*. This value is the **tenantId** property in the output of the following command:

    ```
    az account show
    ```

13. Enter your *Red Hat OpenShift pull secret*. You can download your pull secret from Pull secret.

14. Add your *SSH private key* and *SSH public key* to use to connect to the cluster. You can use an existing key pair, or create a new pair using a key generation program. See Generating an SSH private key and adding it to the agent for more information about how to generate a key.

15. Click Create. When you create the provider connection, it is added to the list of provider connections.

You can create a cluster that uses this provider connection by completing the steps in Creating a cluster on Microsoft Azure.

6.2.3. Deleting your provider connection

When you are no longer managing a cluster that is using a provider connection, delete the provider connection to protect the information in the provider connection.

1. From the navigation menu, navigate to Automate infrastructure > Clusters.
2. Select Provider connections.
3. Select the options menu for the provider connection that you want to delete.
4. Select Delete connection.

6.3. CREATING A PROVIDER CONNECTION FOR GOOGLE CLOUD PLATFORM

You need a provider connection to use Red Hat Advanced Cluster Management for Kubernetes console to create and manage a Red Hat OpenShift Container Platform cluster on Google Cloud Platform (GCP).

Note: This procedure is a prerequisite for creating a cluster with Red Hat Advanced Cluster Management for Kubernetes.

6.3.1. Prerequisites

You must have the following prerequisites before creating a provider connection:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster

- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on GCP
- GCP login credentials, which include user Google Cloud Platform Project ID and Google Cloud Platform service account JSON key. See Creating and managing projects.
- Account permissions that allow installing clusters on GCP. See Configuring a GCP project for instructions on how to configure an account.

6.3.2. Creating a provider connection by using the console

To create a provider connection from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the Clusters page, select the *Provider connections* tab.
   Existing provider connections are displayed.
3. Select Add a connection.
4. Select Google Cloud Platform as your provider.
5. Add a name for your provider connection.
6. Select a namespace for your provider connection from the list.

   TIP

   Create a namespace specifically to host your provider connections, for both convenience and security.

7. You can optionally add a *Base DNS domain* for your provider connection. If you add the base DNS domain to the provider connection, it is automatically populated in the correct field when you create a cluster with this provider connection.
8. Add your *Google Cloud Platform project ID* for your GCP account. Log in to GCP to retrieve your settings.
9. Add your *Google Cloud Platform service account JSON key*. Complete the following steps to create one with the correct permissions:
   a. In the GCP main menu, select IAM & Admin and start the Service Accounts applet
   b. Select Create Service Account
   c. Provide the *Name*, *Service account ID*, and *Description* of your service account.
   d. Select Create to create the service account.
   e. Select a role of Owner, and click Continue.
   f. Click Create Key
   g. Select JSON, and click Create.

   h. Save the resulting file to your computer.

   i. Provide the contents for the *Google Cloud Platform service account JSON key.*

10. Enter your *Red Hat OpenShift pull secret*. You can download your pull secret from Pull secret.

11. Add your *SSH private key* and *SSH public key* so you can access the cluster. You can use an existing key pair, or create a new pair using a key generation program. See Generating an SSH private key and adding it to the agent for more information about how to generate a key.

12. Click Create. When you create the provider connection, it is added to the list of provider connections.

You can use this connection when you create a cluster by completing the steps in Creating a cluster on Google Cloud Platform.

6.3.3. Deleting your provider connection

When you are no longer managing a cluster that is using a provider connection, delete the provider connection to protect the information in the provider connection.

1. From the navigation menu, navigate to Automate infrastructure > Clusters.
2. Select Provider connections.
3. Select the options menu beside the provider connection that you want to delete.
4. Select Delete connection.

6.4. CREATING A PROVIDER CONNECTION FOR VMWARE VSPHERE

You need a provider connection to use Red Hat Advanced Cluster Management for Kubernetes console to deploy and manage a Red Hat OpenShift Container Platform cluster on VMware vSphere. Note: Only OpenShift Container Platform versions 4.5.x and later, are supported.

Note: This procedure must be done before you can create a cluster with Red Hat Advanced Cluster Management.

6.4.1. Prerequisites

You must have the following prerequisites before you create a provider connection:

- A deployed Red Hat Advanced Cluster Management hub cluster on OpenShift Container Platform version 4.5, or later.
- Internet access for your Red Hat Advanced Cluster Management hub cluster so it can create the Kubernetes cluster on VMware vSphere.
- VMware vSphere login credentials and vCenter requirements configured for OpenShift Container Platform when using installer-provisioned infrastructure. See Installing a cluster on vSphere. These credentials include the following information:
   - vCenter account privileges.

- Cluster resources.
- DHCP available.
- ESXi hosts have time synchronized (for example, NTP).

6.4.2. Creating a provider connection by using the console

To create a provider connection from the Red Hat Advanced Cluster Management console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, select the*Provider connections* tab.
   Existing provider connections are displayed.
3. Select Add a connection.
4. Select VMware vSphere as your provider.
5. Add a name for your provider connection.
6. Select a namespace for your provider connection from the list.
   Tip: Create a namespace specifically to host your provider connections for both convenience and added security.
7. You can optionally add a Base DNS domain for your provider connection. If you add the base DNS domain to the provider connection, it is automatically populated in the correct field when you create a cluster with this provider connection.
8. Add your VMware vCenter server fully-qualified host name or IP address. The value must be defined in the vCenter server root CA certificate. If possible, use the fully-qualified host name.
9. Add your VMware vCenter username.
10. Add your VMware vCenter password.
11. Add your VMware vCenter root CA certificate.
    a. You can download your certificate in the **download.zip** package with the certificate from your VMware vCenter server at: **https://<vCenter_address>/certs/download.zip**. Replace **vCenter_address** with the address to your vCenter server.
    b. Unpackage the **download.zip** file.
    c. Use the certificate from the **certs/<platform>** directory that has a**.0** extension. Tip: You can use the **ls certs/<platform>** command to list all of the available certificates for your platform.
       Replace **platform** with the abbreviation for your platform:**lin**, **mac**, or **win**.

       For example: **certs/lin/3a343545.0**
12. Add your VMware vSphere cluster name.
13. Add your VMware vSphere datacenter.

14. Add your VMware vSphere default datastore.

15. Enter your OpenShift Container Platform pull secret. You can download your pull secret from Pull secret.

16. Add your SSH private key and your SSH public key, which allows you to connect to the cluster. You can use an existing key pair, or create a new one with key generation program. See Generating an SSH private key and adding it to the agent for more information.

17. Click Create. When you create the provider connection, it is added to the list of provider connections.

You can create a cluster that uses this provider connection by completing the steps in Creating a cluster on VMware vSphere.

6.4.3. Deleting your provider connection

When you are no longer managing a cluster that is using a provider connection, delete the provider connection to protect the information in the provider connection.

1. From the navigation menu, navigate to Automate infrastructure >Clusters.

2. Select Provider connections.

3. Select the options menu for the provider connection that you want to delete.

4. Select Delete connection.

6.5. CREATING A PROVIDER CONNECTION FOR BARE METAL

You need a provider connection to use Red Hat Advanced Cluster Management for Kubernetes console to deploy and manage a Red Hat OpenShift Container Platform cluster in a bare metal environment.

6.5.1. Prerequisites

You need the following prerequisites before creating a provider connection:

- A Red Hat Advanced Cluster Management for Kubernetes hub cluster that is deployed. When managing bare metal clusters, you must have the hub cluster installed on Red Hat OpenShift Container Platform version 4.5, or later.

- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on your bare metal server.

- Your bare metal server login credentials, which include the libvirt URI, SSH Private Key, and a list of SSH known hosts; see Generating an SSH private key and adding it to the agent

- For a disconnected environment, you must have a configured mirror registry where you can copy the release images for your cluster creation. See Mirroring images for a disconnected installation in the OpenShift Container Platform documentation for more information.

- Account permissions that allow installing clusters on the bare metal infrastructure

6.5.2. Preparing a provisioning host

When you create a bare metal credential and cluster, you must have a provisioning host. The provisioning host is an available bootstrap host VM for the installation. This can be a VM or a service running Kernel-based virtual machine (KVM). You need the details of this host when you are creating the credential and the cluster. Complete the following steps to configure a provisioner host:

1. Log in to the provisioner node using **SSH**.
2. Create a non-root user (user-name) and provide that user with sudo privileges by running the following commands:

   ```
   useradd <user-name>
   passwd <password>
   echo "<user-name> ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/<user-name>
   chmod 0440 /etc/sudoers.d/<user-name>
   ```

3. Create an SSH key for the new user by entering the following command:

   ```
   su - <user-name> -c "ssh-keygen -t rsa -f /home/<user-name>/.ssh/id_rsa -N ""
   ```

4. Log in as the new user on the provisioner node.

   ```
   su - <user-name>
   [user-name@provisioner ~]$
   ```

5. Use Red Hat Subscription Manager to register the provisioner node by entering the following commands:

   ```
   sudo subscription-manager register --username=<user-name> --password=<password> --
   auto-attach
   sudo subscription-manager repos --enable=rhel-8-for-x86_64-appstream-rpms --enable=rhel-
   8-for-x86_64-baseos-rpms
   ```

   For more information about Red Hat Subscription Manager, see Using and Configuring Red Hat Subscription Manager in the Red Hat OpenShift Container Platform documentation.

6. Install required packages by running the following command:

   ```
   sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
   ```

7. Modify the user to add the **libvirt** group to the newly created user.

   ```
   sudo usermod --append --groups libvirt <user-name>
   ```

8. Restart **firewalld** and enable the **http** service by entering the following commands:

   ```
   sudo systemctl start firewalld
   sudo firewall-cmd --zone=public --add-service=http --permanent
   sudo firewall-cmd --add-port=5000/tcp --zone=libvirt  --permanent
   sudo firewall-cmd --add-port=5000/tcp --zone=public   --permanent
   sudo firewall-cmd --reload
   ```

9. Start and enable the **libvirtd** service by entering the following commands:

```
sudo systemctl start libvirtd
sudo systemctl enable libvirtd --now
```

10. **Create the default storage pool and start it by entering the following commands:**

```
sudo virsh pool-define-as --name default --type dir --target /var/lib/libvirt/images
sudo virsh pool-start default
sudo virsh pool-autostart default
```

11. **View the following examples to configure networking:**

- ***Provisioning Network (IPv4 address)***

```
sudo nohup bash -c """
    nmcli con down "$PROV_CONN"
    nmcli con delete "$PROV_CONN"
    # RHEL 8.1 appends the word "System" in front of the connection, delete in case it
exists
    nmcli con down "System $PROV_CONN"
    nmcli con delete "System $PROV_CONN"
    nmcli connection add ifname provisioning type bridge con-name provisioning
    nmcli con add type bridge-worker ifname "$PROV_CONN" master provisioning
    nmcli connection modify provisioning ipv4.addresses 172.22.0.1/24 ipv4.method
manual
    nmcli con down provisioning
    nmcli con up provisioning"""
```

The SSH connection might disconnect after you complete this step.

The IPv4 address can be any address as long as it is not routable using the baremetal network.

- ***Provisioning Network (IPv6 address)***

```
sudo nohup bash -c """
    nmcli con down "$PROV_CONN"
    nmcli con delete "$PROV_CONN"
    # RHEL 8.1 appends the word "System" in front of the connection, delete in case it
exists
    nmcli con down "System $PROV_CONN"
    nmcli con delete "System $PROV_CONN"
    nmcli connection add ifname provisioning type bridge con-name provisioning
    nmcli con add type bridge-worker ifname "$PROV_CONN" master provisioning
    nmcli connection modify provisioning ipv6.addresses fd00:1101::1/64 ipv6.method
manual
    nmcli con down provisioning
    nmcli con up provisioning"""
```

The SSH connection might disconnect after you complete this step.

The IPv6 address can be any address as long as it is not routable using the baremetal network.

Ensure that UEFI is enabled and UEFI PXE settings are set to the IPv6 protocol when using IPv6 addressing.

12. Reconnect to the provisioner node by using **ssh** (if required).

```
# ssh <user-name>@provisioner.<cluster-name>.<domain>
```

13. Verify the connection bridges have been correctly created by running the following command:

```
nmcli con show
```

Your returned results resemble the following content:

NAME	UUID	TYPE	DEVICE
baremetal	4d5133a5-8351-4bb9-bfd4-3af264801530	bridge	baremetal
provisioning	43942805-017f-4d7d-a2c2-7cb3324482ed	bridge	provisioning
virbr0	d9bca40f-eee1-410b-8879-a2d4bb0465e7	bridge	virbr0
bridge-worker-eno1	76a8ed50-c7e5-4999-b4f6-6d9014dd0812	ethernet	eno1
bridge-worker-eno2	f31c3353-54b7-48de-893a-02d2b34c4736	ethernet	eno2

14. Create a **pull-secret.txt** file by completing the following steps:

```
vim pull-secret.txt
```

a. In a web browser, navigate to Install OpenShift on Bare Metal with user-provisioned infrastructure, and scroll down to the*Downloads* section.

b. Click Copy pull secret.

c. Paste the contents into the **pull-secret.txt** file and save the contents in the home directory of the **user-name** user.

You are ready to create your bare metal credential.

6.5.3. Creating a provider connection by using the console

To create a provider connection from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, select the*Provider connections* tab.
   Existing provider connections are displayed.
3. Select Add connection.
4. Select Bare metal as your provider.
5. Add a name for your provider connection.
6. Select a namespace for your provider connection from the list.
   Tip: Create a namespace specifically to host your provider connections, both for convenience and added security.
7. You can optionally add a *Base DNS domain* for your provider connection. If you add the base DNS domain to the provider connection, it is automatically populated in the correct field when you create a cluster with this provider connection.
8. Add your *libvirt URI*. The libvirt URI is for your provisioning node that you created for your bootstrap node. Your libvirt URI should resemble the following example:

   ```
   <qemu+ssh>:://<user-name>@<provision-host.com>/system
   ```

   - Replace **qemu+ssh** with your method of connecting to the libvirt daemon on the provisioning host.
   - Replace **user-name** with the user name that has access to create the bootstrap node on the provisioning host.
   - Replace **provision-host.com** with a link to your provisioning host. This can be either an IP address or a fully-qualified domain name address.
     See Connection URIs for more information.
9. Add a list of your SSH known hosts for the provisioning host. This value can be an IP address or a fully-qualified domain name address, but is best to use the same format that you used in the libvirt URI value.
10. Enter your *Red Hat OpenShift pull secret*. You can download your pull secret fromPull secret.
11. Add your *SSH private key* and your*SSH public key* so you can access the cluster. You can use an existing key, or use a key generation program to create a new one. See Generating an SSH private key and adding it to the agent for more information about how to generate a key.
12. For disconnected installations only: Complete the fields in the Configuration for disconnected installation subsection with the required information:
    - *Image registry mirror*: This value contains the disconnected registry path. The path contains the hostname, port, and repository path to all of the installation images for disconnected installations. Example: **repository.com:5000/openshift/ocp-release**.

The path creates an image content source policy mapping in the **install-config.yaml** to the Red Hat OpenShift Container Platform release images. As an example, **repository.com:5000** produces this **imageContentSource** content:

```
imageContentSources:
- mirrors:
  - registry.example.com:5000/ocp4
  source: quay.io/openshift-release-dev/ocp-release-nightly
- mirrors:
  - registry.example.com:5000/ocp4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - registry.example.com:5000/ocp4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

- *Bootstrap OS image*: This value contains the URL to the image to use for the bootstrap machine.
- *Cluster OS image*: This value contains the URL to the image to use for Red Hat OpenShift Container Platform cluster machines.
- *Additional trust bundle*: This value provides the contents of the certificate file that is required to access the mirror registry.
  Note: If you are deploying managed clusters from a hub that is in a disconnected environment, and want them to be automatically imported post install, add an Image Content Source Policy to the **install-config.yaml** file by using the **YAML** editor. A sample entry is shown in the following example:

```
imageContentSources:
- mirrors:
  - registry.example.com:5000/rhacm2
  source: registry.redhat.io/rhacm2
```

13. Click Create. When you create the provider connection, it is added to the list of provider connections.

You can create a cluster that uses this provider connection by completing the steps in Creating a cluster on bare metal.

6.5.4. Deleting your provider connection

When you are no longer managing a cluster that is using a provider connection, delete the provider connection to protect the information in the provider connection.

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. Select Provider connections.
3. Select the options menu beside the provider connection that you want to delete.
4. Select Delete connection.

CHAPTER 7. CREATING A CLUSTER

Learn how to create Red Hat OpenShift Container Platform clusters across cloud providers with Red Hat Advanced Cluster Management for Kubernetes.

- Creating a cluster on Amazon Web Services
- Creating a cluster on Google Cloud Platform
- Creating a cluster on Microsoft Azure
- Creating a cluster on VMware vSphere
- Creating a cluster on bare metal

7.1. CREATING A CLUSTER ON AMAZON WEB SERVICES

You can use the Red Hat Advanced Cluster Management for Kubernetes console to create a Red Hat OpenShift Container Platform cluster on Amazon Web Services (AWS).

7.1.1. Prerequisites

You must have the following prerequisites before creating a cluster on AWS:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on Amazon Web Services
- AWS provider connection. See Creating a provider connection for Amazon Web Services for more information.
- A configured domain in AWS. See Configuring an AWS accountfor instructions on how to configure a domain.
- Amazon Web Services (AWS) login credentials, which include user name, password, access key ID, and secret access key. See Understanding and Getting Your Security Credentials.
- A OpenShift Container Platform image pull secret. See Using image pull secrets.

Note: If you change your cloud provider access key, you must manually update the provisioned cluster access key. For more information, see the known issue, Automatic secret updates for provisioned clusters is not supported.

7.1.2. Creating your cluster with the Red Hat Advanced Cluster Management for Kubernetes console

To create clusters from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the Clusters page, Click Add Cluster.
3. Select Create a cluster.

Note: This procedure is for creating a cluster. If you have an existing cluster that you want to import, see Importing a target managed cluster to the hub clusterfor those steps.

4. Enter a name for your cluster. This name is used in the hostname of the cluster.
   Tip: You can view the**yaml** content updates as you enter the information in the console by setting the *YAML* switch to ON.

5. Select Amazon Web Services for the infrastructure provider.

6. Specify a Release image that you want to use for the cluster. This identifies the version of the OpenShift Container Platform image that is used to create the cluster. If the version that you want to use is available, you can select the image from the list of images. If the image that you want to use is not a standard image, you can enter the url to the image that you want to use. See Release images for more information about release images.

7. Select your provider connection from the available connections on the list. If you do not have one configured, or want to configure a new one, select Add connection. See Creating a provider connection for Amazon Web Services for more information about creating a provider connection.

8. Add the *Additional Labels* that you want to associate with your cluster. These labels help to identify the cluster and limit search results.

9. Configure the *Node pools* for your cluster.
   The node pools define the location and size of the nodes that are used for your cluster.

   The *Region* specifies where the nodes are located geographically. A closer region might provide faster performance, but a more distant region might be more distributed.

   - Master pool: There are three Master nodes that are created for your cluster in the master pool. The master nodes share the management of the cluster activity. You can select multiple zones within the region for a more distributed group of master nodes. You can change the type and size of your instance after it is created, but you can also specify it in this section. The default values are *mx5.xlarge - 4 vCPU, 16 GiB RAM - General Purpose* with 100 GiB of root storage.

   - Worker pools: You can create zero or more worker nodes in a worker pool to run the container workloads for the cluster. They can be in a single worker pool, or distributed across multiple worker pools. If zero worker nodes are specified, the master nodes also function as worker nodes.

10. Configure the cluster networking options that are necessary. Enter the base DNS information that you configured for your AWS account. If there is already a base domain associated with the selected provider connection, that value is populated in that field. You can change the value by overwriting it. See Configuring an AWS accountfor more information. This name is used in the hostname of the cluster.

11. Optional: Configure a label for the cluster.

12. Click Create. You can view your cluster details after the create and import process is complete.
    Note: You do not have to run the**kubectl** command that is provided with the cluster details to import the cluster. When you create the cluster, it is automatically configured under the management of Red Hat Advanced Cluster Management.

7.1.3. Accessing your cluster

To access a cluster that is managed by Red Hat Advanced Cluster Management for Kubernetes, complete the following steps:

1. From the Red Hat Advanced Cluster Management navigation menu, navigate to Automate infrastructure >Clusters.
2. Select the name of the cluster that you created or want to access. The cluster details are displayed.
3. Select Reveal credentials to view the user name and password for the cluster. Note these values to use when you log in to the cluster.
4. Select Console URL to link to the cluster.
5. Log in to the cluster by using the user ID and password that you found in step 3.
6. Select Actions >Launch to cluster for the cluster that you want to access.
   Tip: If you already know the login credentials, you can access the cluster by selecting Actions >Launch to cluster for the cluster that you want to access.

7.2. CREATING A CLUSTER ON MICROSOFT AZURE

You can use the Red Hat Advanced Cluster Management for Kubernetes console to deploy a Red Hat OpenShift Container Platform cluster on Microsoft Azure.

7.2.1. Prerequisites

You must have the following prerequisites before creating a cluster on Azure:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on Azure
- Azure provider connection. See Creating a provider connection for Microsoft Azurefor more information.
- A configured domain in Azure. See Configuring a custom domain name for an Azure cloud service for instructions on how to configure a domain.
- Azure login credentials, which include user name and password. See azure.microsoft.com.
- Azure service principals, which include **clientId**, **clientSecret**, and **tenantId**. See azure.microsoft.com.
- A OpenShift Container Platform image pull secret. See Using image pull secrets.

Note: If you change your cloud provider access key, you must manually update the provisioned cluster access key. For more information, see the known issue, Automatic secret updates for provisioned clusters is not supported.

7.2.2. Creating your cluster with the Red Hat Advanced Cluster Management for Kubernetes console

To create clusters from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, ClickAdd Cluster.
3. Select Create a cluster.
   Note: This procedure is for creating a cluster. If you have an existing cluster that you want to import, see Importing a target managed cluster to the hub clusterfor those steps.
4. Enter a name for your cluster. This name is used in the hostname of the cluster.
   Tip: You can view the**yaml** content updates as you enter the information in the console by setting the *YAML* switch toON.
5. Select Microsoft Azure for the infrastructure provider.
6. Specify a Release imagethat you want to use for the cluster. This identifies the version of the OpenShift Container Platform image that is used to create the cluster. If the version that you want to use is available, you can select the image from the list of images. If the image that you want to use is not a standard image, you can enter the URL to the image that you want to use. See Release images for more information about release images.
7. Select your provider connection from the available connections on the list. If you do not have one configured, or want to configure a new one, select Add connection. See Creating a provider connection for Microsoft Azure for more information about creating a provider connection.
8. Add the *Additional Labels* that you want to associate with your cluster. These labels help to identify the cluster and limit search results.
9. Configure the *Node pools* for your cluster.
   The node pools define the location and size of the nodes that are used for your cluster.

   The *Region* specifies where the nodes are located geographically. A closer region might provide faster performance, but a more distant region might be more distributed.

   - Master pool: There are three Master nodes that are created for your cluster in the master pool. The master nodes share the management of the cluster activity. You can select multiple zones within the region for a more distributed group of master nodes. You can change the type and size of your instance after it is created, but you can also specify it in this section. The default values are *Standard_D4s_v3 - 4 vCPU, 16 GiB RAM - General Purpose* with 128 GiB of root storage.
   - Worker pools: You can create zero or more worker nodes in a worker pool to run the container workloads for the cluster. They can be in a single worker pool, or distributed across multiple worker pools. If zero worker nodes are specified, the master nodes also function as worker nodes.
10. Configure the cluster networking options that are necessary.
    Enter the base DNS information that you configured for your Azure account. If there is already a base DNS associated with the selected provider connection, that value is populated in that field. You can change the value by overwriting it. See Configuring a custom domain name for an Azure cloud service for more information. This name is used in the hostname of the cluster.
11. Optional: Configure a label for the cluster.
12. Click Create. You can view your cluster details after the create and import process is complete.

Note: You do not have to run the **kubectl** command that is provided with the cluster details to import the cluster. When you create the cluster, it is automatically configured under the management of Red Hat Advanced Cluster Management for Kubernetes.

7.2.3. Accessing your cluster

To access a cluster that is managed by Red Hat Advanced Cluster Management for Kubernetes, complete the following steps:

1. From the Red Hat Advanced Cluster Management for Kubernetes navigation menu, navigate to Automate infrastructure > Clusters.
2. Select the name of the cluster that you created or want to access. The cluster details are displayed.
3. Select Reveal credentials to view the user name and password for the cluster. Note these values to use when you log in to the cluster.
4. Select Console URL to link to the cluster.
5. Log in to the cluster by using the user ID and password that you found in step 3.
6. Select Actions > Launch to cluster for the cluster that you want to access.
   Tip: If you already know the login credentials, you can access the cluster by selecting Actions > Launch to cluster for the cluster that you want to access.

7.3. CREATING A CLUSTER ON GOOGLE CLOUD PLATFORM

Follow the procedure to create a Red Hat OpenShift Container Platform cluster on Google Cloud Platform (GCP). For more information about Google Cloud Platform, see Google Cloud Platform.

7.3.1. Prerequisites

You must have the following prerequisites before creating a cluster on GCP:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster so it can create the Kubernetes cluster on GCP
- GCP provider connection. See Creating a a provider connection for Google Cloud Platform for more information.
- A configured domain in GCP. See Setting up a custom domain for instructions on how to configure a domain.
- GCP login credentials, which include user name and password.
- A OpenShift Container Platform image pull secret. See Using image pull secrets.

Note: If you change your cloud provider access key, you must manually update the provisioned cluster access key. For more information, see the known issue, Automatic secret updates for provisioned clusters is not supported.

7.3.2. Creating your cluster with the Red Hat Advanced Cluster Management for Kubernetes console

To create clusters from the Red Hat Advanced Cluster Management for Kubernetes console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, ClickAdd Cluster.
3. Select Create a cluster.
   Note: This procedure is for creating a cluster. If you have an existing cluster that you want to import, see Importing a target managed cluster to the hub clusterfor those steps.
4. Enter a name for your cluster. This name is used in the hostname of the cluster. There are some restrictions that apply to naming your GCP cluster. These restrictions include not beginning the name with **goog** or containing a group of letters and numbers that resemble **google** anywhere in the name. SeeBucket naming guidelines for the complete list of restrictions.
   Tip: You can view the**yaml** content updates as you enter the information in the console by setting the *YAML* switch toON.
5. Select Google Cloud for the infrastructure provider.
6. Specify a Release image that you want to use for the cluster. This identifies the version of the OpenShift Container Platform image that is used to create the cluster. If the version that you want to use is available, you can select the image from the list of images. If the image that you want to use is not a standard image, you can enter the URL to the image that you want to use. See Release images for more information about release images.
7. Select your provider connection from the available connections on the list. If you do not have one configured, or want to configure a new one, select Add connection. See Creating a provider connection for Google Cloud Platform for more information about creating a provider connection.
8. Add the *Additional Labels* that you want to associate with your cluster. These labels help to identify the cluster and limit search results.
9. Configure the *Node pools* for your cluster.
   The node pools define the location and size of the nodes that are used for your cluster.

   The *Region* specifies where the nodes are located geographically. A closer region might provide faster performance, but a more distant region might be more distributed.

   - Master pool: There are three Master nodes that are created for your cluster in the master pool. The master nodes share the management of the cluster activity. You can select multiple zones within the region for a more distributed group of master nodes. You can change the type and size of your instance after it is created, but you can also specify it in this section. The default values are *n1-standard-1 - n1-standard-1 1 vCPU - General Purpose* with 500 GiB of root storage.
   - Worker pools: You can create zero or more worker nodes in a worker pool to run the container workloads for the cluster. They can be in a single worker pool, or distributed across multiple worker pools. If zero worker nodes are specified, the master nodes also function as worker nodes.
10. Configure the cluster networking options that are necessary.

Enter the base DNS information that you configured for your Google Cloud Platform account. If there is already a base DNS that is associated with the selected provider connection, that value is populated in that field. You can change the value by overwriting it. See Setting up a custom domainfor more information. This name is used in the hostname of the cluster.

11. Optional: Configure a label for the cluster.

12. Click Create.

You can view your cluster details after the create and import process is complete.

+ Note: You do not have to run the**kubectl** command that is provided with the cluster details to import the cluster. When you create the cluster, it is automatically configured under the management of Red Hat Advanced Cluster Management for Kubernetes.

7.3.3. Accessing your cluster

To access a cluster that is managed by Red Hat Advanced Cluster Management for Kubernetes, complete the following steps:

1. From the Red Hat Advanced Cluster Management for Kubernetes navigation menu, navigate to Automate infrastructure >Clusters.

2. Select the name of the cluster that you created or want to access. The cluster details are displayed.

3. Select Reveal credentials to view the user name and password for the cluster. Note these values to use when you log in to the cluster.

4. Select Console URL to link to the cluster.

5. Log in to the cluster by using the user ID and password that you found in step 3.

6. Select Actions >Launch to cluster for the cluster that you want to access.
   Tip: If you already know the login credentials, you can access the cluster by selecting Actions >Launch to cluster for the cluster that you want to access.

7.4. CREATING A CLUSTER ON VMWARE VSPHERE

You can use the Red Hat Advanced Cluster Management for Kubernetes console to deploy a Red Hat OpenShift Container Platform cluster on VMware vSphere.

7.4.1. Prerequisites

You must have the following prerequisites before creating a cluster on vSphere:

- A Red Hat Advanced Cluster Management hub cluster that is deployed on OpenShift Container Platform version 4.5, or later.

- Internet access for your Red Hat Advanced Cluster Management hub cluster so it can create the Kubernetes cluster on vSphere.

- vSphere provider connection. See Creating a provider connection for VMware vSpherefor more information.

- A Red Hat OpenShift image pull secret. See Using image pull secrets.
- The following information for the VMware instance where you are deploying:
  - Required static IP addresses for API and Ingress instances.
  - DNS records for:
    - api.<cluster_name>.<base_domain> which must point to the static API VIP.
    - *.apps.<cluster_name>.<base_domain> which must point to the static IP address for Ingress VIP.

7.4.2. Creating your cluster with the Red Hat Advanced Cluster Management for Kubernetes console

To create clusters from the Red Hat Advanced Cluster Management console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, clickAdd Cluster.
3. Select Create a cluster.
   Note: This procedure is for creating a cluster. If you have an existing cluster that you want to import, see Importing a target managed cluster to the hub clusterfor those steps.
4. Enter a name for your cluster. This name is used in the hostname of the cluster.
   Note: This value must match the name that you used to create the DNS records listed in the provider connection prerequisites section.

   Tip: You can view the**yaml** content updates as you enter the information in the console by setting the *YAML* switch to ON.
5. Select VMware vSphere for the infrastructure provider.
6. Specify a Release image that you want to use for the cluster. This identifies the version of the OpenShift Container Platform image that is used to create the cluster. If the version that you want to use is available, you can select the image from the list of images. If the image that you want to use is not a standard image, you can enter the URL of the image that you want to use. See Release images for more information.Note: Only release images for OpenShift Container Platform versions 4.5.x and higher are supported.
7. Select your provider connection from the available connections on the list. If you do not have one configured, or want to configure a new one, select Add connection. See Creating a provider connection for more information about creating a provider connection.
8. Enter the base domain information that you configured for your vSphere account. If there is already a base domain associated with the selected provider connection, that value is populated in that field. You can change the value by overwriting it. Note: This value must match the name that you used to create the DNS records listed in the prerequisites section. This name is used in the hostname of the cluster.
9. Add the *Additional Labels* that you want to associate with your cluster. These labels help to identify the cluster and limit search results.
10. Configure the *Node pools* for your cluster.

The node pools define the location and size of the nodes that are used for your cluster.

You can create one or more worker nodes in a worker pool to run the container workloads for the cluster. They can be in a single worker pool, or distributed across multiple worker pools.

11. Configure the cluster networking options, which are shown in the following list:
    a. vSphere network name - The VMware vSphere network name.
    b. API VIP - The IP address to use for internal API communication. Note: This value must match the name that you used to create the DNS records listed in the prerequisites section. If not provided, the DNS must be pre-configured so that **api.** resolves correctly.
    c. Ingress VIP - The IP address to use for ingress traffic. Note: This value must match the name that you used to create the DNS records listed in the prerequisites section. If not provided, the DNS must be pre-configured so that **test.apps.** resolves correctly.
12. Optional: Configure a label for the cluster.
13. Click Create. You can view your cluster details after the create and import process is complete.
    Note: When you create the cluster, it is automatically configured under the management of Red Hat Advanced Cluster Management. You do not have to run the **kubectl** command that is provided with the cluster details to import the cluster.

7.4.3. Accessing your cluster

To access a cluster that is managed by Red Hat Advanced Cluster Management, complete the following steps:

1. If you already know the log in credentials, you can access the cluster by selecting the *Options* menu for the cluster, and selectingLaunch to cluster.
2. If you do not know the log in credentials
    a. From the Red Hat Advanced Cluster Management navigation menu, navigate to Automate infrastructure >Clusters.
    b. Select the name of the cluster that you created or want to access. The cluster details are displayed.
    c. Select Reveal credentials to view the user name and password for the cluster. Use these values when you log in to the cluster.
3. Select Console URL to link to the cluster.
4. Log in to the cluster by using the user ID and password that you found in step 3.
5. Select Actions >Launch to cluster for the cluster that you want to access.
   Tip: If you already know the login credentials, you can access the cluster by selecting Actions >Launch to cluster for the cluster that you want to access.

7.5. CREATING A CLUSTER ON BARE METAL

You can use the Red Hat Advanced Cluster Management for Kubernetes console to create a Red Hat OpenShift Container Platform cluster in a bare metal environment.

7.5.1. Prerequisites

You need the following prerequisites before creating a cluster in a bare metal environment:

- A deployed Red Hat Advanced Cluster Management for Kubernetes hub cluster on OpenShift Container Platform version 4.5, or later.
- Internet access for your Red Hat Advanced Cluster Management for Kubernetes hub cluster (connected) or a connection to an internal or mirror registry that has a connection to the Internet (disconnected) to retrieve the required images for creating the cluster.
- A temporary external KVM host that runs a bootstrap virtual machine, which is used to create a Hive cluster. See Preparing a provisioning host for more information.
- Your bare metal server login credentials, which includes the libvirt URI from the bootstrap virtual machine in the previous item, the SSH Private Key, and a list of SSH known hosts. See Setting up the environment for an OpenShift installationfor more information.
- Bare metal provider connection; see Creating a provider connection for bare metalfor more information.
- Login credentials for your bare metal environment, which include user name, password, and Baseboard Management Controller Address.
- A configured bare metal asset, if you are are enabling certificate verification. See Creating and modifying bare metal assets for more information.
- A OpenShift Container Platform image pull secret; see Using image pull secrets.
  Notes:
  - The bare metal asset, managed bare metal cluster, and its related secret must be in the same namespace.
  - If you change your cloud provider access key, you must manually update the provisioned cluster access key. For more information, see the known issue, Automatic secret updates for provisioned clusters is not supported.

7.5.2. Creating your cluster with the Red Hat Advanced Cluster Management console

To create clusters from the Red Hat Advanced Cluster Management console, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. On the *Clusters* page, ClickAdd Cluster.
3. Select Create a cluster.
   Note: This procedure is for creating a cluster. If you have an existing cluster that you want to import, see Importing a target managed cluster to the hub clusterfor those steps.

4. Enter a name for your cluster. For a bare metal cluster, this name cannot be an arbitrary name. It is associated with the cluster URL. Make sure that the cluster name that you use is consistent with your DNS and network setup.
   Tip: You can view the**yaml** content updates as you enter the information in the console by setting the *YAML* switch to ON.

5. Select Bare Metal for the infrastructure provider.

6. Specify a Release image that you want to use for the cluster. This identifies the version of the Red Hat OpenShift Container Platform image that is used to create the cluster. If the version that you want to use is available, you can select the image from the list of images. If the image that you want to use is not a standard image, you can enter the URL to the image that you want to use. See Release images for more information about release images.

7. Select your provider connection from the available connections on the list. If you do not have one configured, or want to configure a new one, select Add connection. See Creating a provider connection for bare metal for more information about creating a provider connection.

8. Optional: Configure additional labels for the cluster.

9. Select your hosts from the list of hosts that are associated with your provider connection. Select a minimum of three assets that are on the same bridge networks as the hypervisor. The list of hosts is compiled from the existing Bare Metal Assets. If you do not have any Bare Metal Assets created, then you can create or import them before you continue with the creation process. Alternatively, you can select Disable certificate verification to bypass the requirement.

10. Configure the cluster networking options.

Parameter	Description	Required or Optional
Base DNS domain	The base domain of your provider, which is used to create routes to your Red Hat OpenShift Container Platform cluster components. It is configured in your cluster provider’s DNS as a Start of Authority (SOA) record. This setting cannot be changed after the cluster is created.	Required

Parameter	Description	Required or Optional
Network type	The pod network provider plug-in to deploy. Only the OpenShiftSDN plug-in is supported on OpenShift Container Platform 4.3. The OVNKubernetes plug-in is available as a technical preview on OpenShift Container Platform versions 4.3, 4.4, and 4.5. It is generally available on OpenShift Container Platform version 4.6, and later. OVNKubernetes must be used with IPv6. The default value is **OpenShiftSDN**.	Required
Cluster network 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. The default values is 10.128.0.0/14.	Required
Network host prefix	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. The default is 23.	Required
Service network CIDR	A block of IP addresses for services. OpenShiftSDN allows only one serviceNetwork block. The address must not overlap any other network block. The default value is 172.30.0.0/16.	Required
Machine CIDR	A block of IP addresses used by the OpenShift Container Platform hosts. The address block must not overlap any other network block. The default value is 10.0.0.0/16.	Required

Parameter	Description	Required or Optional
Provisioning network CIDR	The CIDR for the network to use for provisioning. The example format is: 172.30.0.0/16.	Required
Provisioning network interface	The name of the network interface on the control plane nodes that are connected to the provisioning network.	Required
Provisioning network bridge	The name of the bridge on the hypervisor that is attached to the provisioning network.	Required
External network bridge	The name of the bridge of the hypervisor that is attached to the external network.	Required
DNS VIP	The Virtual IP to use for internal DNS communication. This parameter only applies to OpenShift Container Platform versions 4.4, and earlier.	Required for OpenShift Container Platform versions 4.4, and earlier.
API VIP	The Virtual IP to use for internal API communication. The DNS must be pre-configured with an A/AAAA or CNAME record so the **api.<cluster_name>.<Base DNS domain>** path resolves correctly.	Required
Ingress VIP	The Virtual IP to use for ingress traffic. The DNS must be pre-configured with an A/AAAA or CNAME record so the ***.apps.<cluster_name>.<Base DNS domain>** path resolves correctly.	Optional

11. **Optional: Update the advanced settings, if you want to change the setting for including a configmap.**

12. **Click Create. You can view your cluster details after the create and import process is complete.**
    Note: You do not have to run thekubectl command that is provided with the cluster details to import the cluster. When you create the cluster, it is automatically configured under the management of Red Hat Advanced Cluster Management for Kubernetes.

7.5.3. Accessing your cluster

To access a cluster that is managed by Red Hat Advanced Cluster Management for Kubernetes, complete the following steps:

1. From the Red Hat Advanced Cluster Management for Kubernetes navigation menu, navigate to Automate infrastructure >Clusters.

2. Select the name of the cluster that you created or want to access. The cluster details are displayed.

3. Select Reveal credentials to view the user name and password for the cluster. Note these values to use when you log in to the cluster.

4. Select Console URL to link to the cluster.

5. Log in to the cluster by using the user ID and password that you found in step 3.

6. Select Actions >Launch to cluster for the cluster that you want to access.
   Tip: If you already know the login credentials, you can access the cluster by selecting Actions >Launch to cluster for the cluster that you want to access.

CHAPTER 8. IMPORTING A TARGET MANAGED CLUSTER TO THE HUB CLUSTER

You can import clusters from different Kubernetes cloud providers. After you import, the targeted cluster becomes a managed cluster for the Red Hat Advanced Cluster Management for Kubernetes hub cluster. Unless otherwise specified, complete the import tasks anywhere where you can access the hub cluster and the targeted managed cluster.

A hub cluster cannot manage *any* other hub cluster, but can manage itself. The hub cluster is configured to automatically be imported and self-managed. You do not need to manually import the hub cluster.

However, if you remove a hub cluster and try to import it again, you need to add the **local-cluster:true** label.

Choose from the following instructions to set up your managed cluster, either from the console or from the CLI:

Required user type or access level: Cluster administrator

- Importing an existing cluster with the console
- Importing a managed cluster with the CLI
- Modifying the klusterlet addons settings of your cluster

8.1. IMPORTING AN EXISTING CLUSTER WITH THE CONSOLE

After you install Red Hat Advanced Cluster Management for Kubernetes, you are ready to import a cluster to manage. You can import from both the console and the CLI. Follow this procedure to import from the console. You need your terminal for authentication during this procedure.

- Prerequisites
- Importing a cluster
- Removing a cluster

8.1.1. Prerequisites

- You need a Red Hat Advanced Cluster Management for Kubernetes hub cluster that is deployed. If you are importing bare metal clusters, you must have the hub cluster installed on Red Hat OpenShift Container Platform version 4.5, or later.
- You need a cluster that you want to manage and Internet connectivity.
- Install **kubectl**. To install **kubectl**, see *Install and Set Up kubectl* in the Kubernetes documentation.
- You need the **base64** command line tool.
- *For importing in a Red Hat OpenShift Dedicated enviroment:*
  - You must have the hub cluster deployed in a Red Hat OpenShift Dedicated environment.

- The default permission in Red Hat OpenShift Dedicated is dedicated-admin, but that does not contain all of the permissions to create a namespace. You must have **cluster-admin** permissions to import and manage a cluster with Red Hat Advanced Cluster Management for Kubernetes.

Required user type or access level: Cluster administrator

8.1.2. Importing a cluster

You can import existing clusters from the Red Hat Advanced Cluster Management for Kubernetes console for each of the available cloud providers.

Note: A hub cluster cannot manage a different hub cluster. A hub cluster is set up to automatically import and manage itself, so you do not have to manually import a hub cluster to manage itself.

1. From the navigation menu, hover over Automate infrastructure and click Clusters.
2. Click Add a cluster.
3. Click Import an existing cluster.
4. Provide a name for the cluster. By default, the namespace is used for the cluster name and namespace.
5. Optional: Add any *Additional labels*.
   Note: If you import a Red Hat OpenShift Dedicated cluster and do not specify a vendor by adding a label for **vendor=OpenShiftDedicated**, or if you add a label for **vendor=auto-detect**, a **managed-by=platform** label is automatically added to the cluster. You can use this added label to identify the cluster as a Red Hat OpenShift Dedicated cluster and retrieve the Red Hat OpenShift Dedicated clusters as a group.
6. Click Save import and generate code to generate the command that you use to deploy the **open-cluster-management-agent-addon**. A confirmation message is displayed.
7. Optional: Configure the Cluster API address that is on the cluster details page by configuring the URL that is displayed in the table when you run the **oc get managedcluster** command.
   a. Log in to your hub cluster with an ID that has **cluster-admin** permissions.
   b. Configure your **kubectl** for your targeted managed cluster.
      See Supported clouds to learn how to configure your **kubectl**.
   c. Edit the managed cluster entry for the cluster that you are importing by entering the following command:

      ```
      oc edit managedcluster <cluster-name>
      ```

      Replace ***cluster-name*** with the name of the managed cluster.
   d. Add the **ManagedClusterClientConfigs** section to the **ManagedCluster** spec in the YAML file, as shown in the following example:

      ```
      spec:
        hubAcceptsClient: true
        managedClusterClientConfigs:
      ```

```
- url: https://multicloud-console.apps.new-managed.dev.redhat.com
```

Replace the value of the URL with the URL that provides external access to the managed cluster that you are importing.

8. In the *Import an existing cluster* window, select Copy command to copy the generated command and token to the clipboard.
   Important: The command contains pull secret information that is copied to each of the imported clusters. Anyone who can access the imported clusters can also view the pull secret information. Consider creating a secondary pull secret at https://cloud.redhat.com/ or create a service account to protect your personal credentials. See Using image pull secrets or Understanding and creating service accounts for more information about pull secrets.

9. Log in to the managed cluster that you want to import.

10. For the Red Hat OpenShift Dedicated environment only: Complete the following steps:

    a. Create the **open-cluster-management-agent** and **open-cluster-management** namespaces or projects on the managed cluster.

    b. Find the klusterlet operator in the OpenShift Container Platform catalog.

    c. Install it in the **open-cluster-management** namespace or project that you created.
       Important: Do not install the operator in the **open-cluster-management-agent** namespace.

    d. Extract the bootstrap secret from the import command by completing the following steps:

       i. Generate the import command:

          A. Select Automate infrastructure > Clusters from the Red Hat Advanced Cluster Management console main navigation.

          B. Select Add a cluster > Import an existing cluster.

          C. Add the cluster information, and select Save import and generate code.

       ii. Copy the import command.

       iii. Paste the import command into a file that you create named **import-command**.

       iv. Run one of the following commands to decode the import command, depending on your version of Red Hat Advanced Cluster Management version 2.2.x:

          - If you are running an upgraded z-stream version of Red Hat Advanced Cluster Management version 2.2.x (for example, 2.2.2), run the following command:

            ```
            cat import-command | awk '{split($0,a,"&&"); print a[3]}' | awk '{split($0,a,"|"); print
            a[1]}' | sed -e "s/^ echo //" | sed 's/\"//g' | base64 -d
            ```

          - If your version of Red Hat Advanced Cluster Management version 2.2 was not upgraded to a later z-stream (for example, you are running version 2.2.0), run the following command:

```
cat import-command | awk '{split($0,a,"&&"); print a[3]}' | awk '{split($0,a,"|"); print
a[1]}' | sed -e "s/^ echo //" | base64 -d
```

v. Find and copy the secret with the name **bootstrap-hub-kubeconfig** in the output.

vi. Apply the secret to the **open-cluster-management-agent** namespace on the managed cluster.

vii. Create the klusterlet resource using the example in the installed operator, the clusterName should be changed the same name as cluster name that was set during the import.
Note: When the**managedcluster** resource is successfully registered to the hub, there are two klusterlet operators installed. One klusterlet operator is in the **open-cluster-management** namespace, and the other is in the**open-cluster-management-agent** namespace. Multiple operators does not affect the function of the klusterlet.

11. For cluster imports that are not in the Red OpenShift Dedicated environmentComplete the following steps:

    a. If necessary, configure your **kubectl** commands for your managed cluster.
    See Supported clouds to learn how to configure your**kubectl** command line interface.

    b. To deploy the **open-cluster-management-agent-addon** to the managed cluster, run the command and token that you copied.

12. Select View cluster to view a summary of your cluster in the*Overview* page.

Your cluster is imported. You can import another by selecting Import another.

8.1.3. Removing an imported cluster

Complete the following procedure to remove an imported cluster and the **open-cluster-management-agent-addon** that was created on the managed cluster.

1. From the *Clusters* page, find your imported cluster in the table.

2. Click Actions >Detach cluster to remove your cluster from management.

Note: If you attempt to detach the hub cluster, which is named**local-cluster**, be aware that the default setting of **disableHubSelfManagement** is **false**. This setting causes the hub cluster to reimport itself and manage itself when it is detached and it reconciles the **MultiClusterHub** controller. It might take hours for the hub cluster to complete the detachment process and reimport. If you want to reimport the hub cluster without waiting for the processes to finish, you can enter the following command to restart the **multiclusterhub-operator** pod and reimport faster:

```
oc delete po -n open-cluster-management `oc get pod -n open-cluster-management | grep
multiclusterhub-operator| cut -d' ' -f1`
```

You can change the value of the hub cluster to not import automatically by changing the **disableHubSelfManagement** value to**true**, as described inInstalling while connected online.

8.2. IMPORTING A MANAGED CLUSTER WITH THE CLI

After you install Red Hat Advanced Cluster Management for Kubernetes, you are ready to import a cluster to manage. You can import from both the console and the CLI. Follow this procedure to import from the CLI.

- Prerequisites
- Supported architecture
- Importing the klusterlet

Important: A hub cluster cannot manage a different hub cluster. A hub cluster is set up to automatically import and manage itself. You do not have to manually import a hub cluster to manage itself.

However, if you remove a hub cluster and try to import it again, you need to add the **local-cluster:true** label.

8.2.1. Prerequisites

- You need a Red Hat Advanced Cluster Management for Kubernetes hub cluster that is deployed. If you are importing bare metal clusters, you must have the hub cluster installed on Red Hat OpenShift Container Platform version 4.5, or later.
- You need a separate cluster that you want to manage and Internet connectivity.
- You need the Red Hat OpenShift Container Platform CLI version 4.5, or later, to run **oc** commands. See Getting started with the OpenShift CLI for information about installing and configuring the Red Hat OpenShift CLI, **oc**.
- You need to install the Kubernetes CLI, **kubectl**. To install **kubectl**, see *Install and Set Up kubectl* in the Kubernetes documentation.
  Note: Download the installation file for CLI tools from the console.

8.2.2. Supported architectures

- Linux (x86_64, s390x)
- macOS

8.2.3. Prepare for import

1. Log in to your *hub cluster*. Run the following command:

   ```
   oc login
   ```

2. Run the following command on the hub cluster to create the namespace. Note: The cluster name that is defined in **<cluster_name>** is also used as the cluster namespace in the**yaml** file file and commands:

   ```
   oc new-project ${CLUSTER_NAME}
   oc label namespace ${CLUSTER_NAME} cluster.open-cluster-
   management.io/managedCluster=${CLUSTER_NAME}
   ```

3. Edit the example ManagedCluster with the following sample of YAML:

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  name: <cluster_name>
spec:
  hubAcceptsClient: true
```

4. Save the file as **managed-cluster.yaml**.

5. Apply the YAML file with the following command:

```
oc apply -f managed-cluster.yaml
```

6. Create the klusterlet addon configuration file. Enter the following example YAML:

```
apiVersion: agent.open-cluster-management.io/v1
kind: KlusterletAddonConfig
metadata:
  name: <cluster_name>
  namespace: <cluster_name>
spec:
  clusterName: <cluster_name>
  clusterNamespace: <cluster_name>
  applicationManager:
    enabled: true
  certPolicyController:
    enabled: true
  clusterLabels:
    cloud: auto-detect
    vendor: auto-detect
  iamPolicyController:
    enabled: true
  policyController:
    enabled: true
  searchCollector:
    enabled: true
  version: 2.2.0
```

7. Save the file as **klusterlet-addon-config.yaml**.

8. Apply the YAML. Run the following command:

```
oc apply -f klusterlet-addon-config.yaml
```

Note: If you import a Red Hat OpenShift Dedicated cluster and do not specify a vendor by adding a label for **vendor=OpenShiftDedicated**, or if you add a label for**vendor=auto-detect**, a **managed-by=platform** label is automatically added to the cluster. You can use this added label to identify the cluster as a Red Hat OpenShift Dedicated cluster and retrieve the Red Hat OpenShift Dedicated clusters as a group.

The ManagedCluster-Import-Controller will generate a secret named **${CLUSTER_NAME}-import**. The **${CLUSTER_NAME}-import** secret contains the**import.yaml** that the user applies to a managed cluster to install klusterlet.

8.2.4. Importing the klusterlet

Important: The import command contains pull secret information that is copied to each of the imported clusters. Anyone who can access the imported clusters can also view the pull secret information.

1. Obtain the **klusterlet-crd.yaml** that was generated by the managed cluster import controller.
   Run the following command:

   ```
   oc get secret ${CLUSTER_NAME}-import -n ${CLUSTER_NAME} -o jsonpath=
   {.data.crds\\.yaml} | base64 --decode > klusterlet-crd.yaml
   ```

2. Obtain the **import.yaml** that was generated by the managed cluster import controller. Run the following command:

   ```
   oc get secret ${CLUSTER_NAME}-import -n ${CLUSTER_NAME} -o jsonpath=
   {.data.import\\.yaml} | base64 --decode > import.yaml
   ```

3. Log in to your target *managed* cluster.
4. Apply the **klusterlet-crd.yaml** that was generated in step 1. Run the following command:

   ```
   kubectl apply -f klusterlet-crd.yaml
   ```

5. Apply the **import.yaml** file that was generated in step 2. Run the following command:

   ```
   kubectl apply -f import.yaml
   ```

6. Validate the pod status on the target managed cluster. Run the following command:

   ```
   kubectl get pod -n open-cluster-management-agent
   ```

7. Validate **JOINED** and **AVAILABLE** status for your imported cluster. Run the following command from the *hub* cluster:

   ```
   kubectl get managedcluster ${CLUSTER_NAME}
   ```

8. Addons will be installed after the managed cluster is **AVAILABLE**. Validate the pod status of addons on the target managed cluster. Run the following command:

   ```
   kubectl get pod -n open-cluster-management-agent-addon
   ```

8.3. MODIFYING THE KLUSTERLET ADDONS SETTINGS OF YOUR CLUSTER

You can modify the settings of **klusterlet addon** to change your configuration using the hub cluster.

The **klusterlet addon** controller manages the functions that are enabled and disabled according to the settings in the **klusterletaddonconfigs.agent.open-cluster-management.io** Kubernetes resource.

The following settings can be updated in the **klusterletaddonconfigs.agent.open-cluster-management.io** Kubernetes resource:

Setting name	Value
applicationmanager	**true** or **false**
policyController	**true** or **false**
searchCollector	**true** or **false**
certPolicyController	**true** or **false**
iamPolicyController	**true** or **false**

8.3.1. Modify using the console on the hub cluster

You can modify the settings of the **klusterletaddonconfigs.agent.open-cluster-management.io** resource by using the hub cluster. Complete the following steps to change the settings:

1. Authenticate into the Red Hat Advanced Cluster Management for Kubernetes console of the hub cluster.
2. From the main menu of the hub cluster console, select Search.
3. In the search parameters, enter the following value: **kind:klusterletaddonconfigs**
4. Select the endpoint resource that you want to update.
5. Find the **spec** section and select Edit to edit the content.
6. Modify your settings.
7. Select Save to apply your changes.

8.3.2. Modify using the command line on the hub cluster

You must have access to the <cluster-name> namespace to modify your settings by using the hub cluster. Complete the following steps:

1. Authenticate into the hub cluster.
2. Enter the following command to edit the resource:

```
kubectl edit klusterletaddonconfigs.agent.open-cluster-management.io <cluster-name> -n
<cluster-name>
```

3. Find the **spec** section.
4. Modify your settings, as necessary.

CHAPTER 9. CONFIGURING A SPECIFIC CLUSTER MANAGEMENT ROLE

When you install Red Hat Advanced Cluster Management for Kubernetes, the default configuration provides the **cluster-admin** role on the Red Hat Advanced Cluster Management hub cluster. This permission enables you to create, manage, and import managed clusters on the hub cluster. In some situations, you might want to limit the access to certain managed clusters that are managed by the hub cluster, rather than providing access to all of the managed clusters on the hub cluster.

You can limit access to certain managed clusters by defining a cluster role and applying it to a user or group. Complete the following steps to configure and apply a role:

1. Define the cluster role by creating a YAML file with the following content:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: <clusterrole-name>
rules:
- apiGroups:
  - cluster.open-cluster-management.io
  resources:
  - managedclusters
  resourceNames:
  - <managed-cluster-name>
  verbs:
  - get
  - list
  - watch
  - update
  - delete
  - deletecollection
  - patch
- apiGroups:
  - cluster.open-cluster-management.io
  resources:
  - managedclusters
  verbs:
  - create
- apiGroups:
  - ""
  resources:
  - namespaces
  resourceNames:
  - <managed-cluster-name>
  verbs:
  - create
  - get
  - list
  - watch
  - update
  - delete
  - deletecollection
  - patch
- apiGroups:
```

```
- register.open-cluster-management.io
resources:
- managedclusters/accept
resourceNames:
- <managed-cluster-name>
verbs:
- update
```

Replace ***clusterrole-name*** with the name of the cluster role that you are creating.

Replace ***managed-cluster-name*** with the name of the managed cluster that you want the user to have access to.

2. Apply the **clusterrole** definition by entering the following command:

```
oc apply <filename>
```

Replace ***filename*** with the name of the YAML file that you created in the previous step.

3. Enter the following command to bind the **clusterrole** to a specified user or group:

```
oc adm policy add-cluster-role-to-user <clusterrole-name> <username>
```

Replace ***clusterrole-name*** with the name of the cluster role that you applied in the previous step. Replace ***username*** with the username to which you want to bind the cluster role.

CHAPTER 10. MANAGEDCLUSTERSETS

A **ManagedClusterSet** is a group of managed clusters. With a **ManagedClusterSet**, you can manage access to all of the managed clusters in the group together. You can also create a ManagedClusterSetBinding resource to bind a ManagedClusterSet resource to a namespace.

10.1. CREATING A MANAGEDCLUSTERSET

You can group managed clusters together in a ManagedClusterSet to limit the user access on managed clusters.

Required access: Cluster administrator

A **ManagedClusterSet** is a cluster-scoped resource, so you must have cluster administration permissions for the cluster where you are creating the **ManagedClusterSet**. A managed cluster cannot be included in more than one **ManagedClusterSet**. Complete the following steps to create a **ManagedClusterSet**:

1. Add the following definition of the **ManagedClusterSet** to your **yaml** file:

```
apiVersion: cluster.open-cluster-management.io/v1alpha1
kind: ManagedClusterSet
metadata:
  name: <clusterset1>
```

   Replace *clusterset1* with the name of your **ManagedClusterSet**.

10.2. ADDING CLUSTERS TO A MANAGEDCLUSTERSET

After your **ManagedClusterSet** is created, you must add one or more managed clusters. Complete the following steps to add managed clusters:

1. Ensure that there is an RBAC **ClusterRole** entry that allows you to **Create** on a virtual subresource of **managedclustersets/join**. Without this permission, you cannot assign a managed cluster to a **ManagedClusterSet**.
   If this entry does not exist, add it to your **yaml** file. A sample entry resembles the following content:

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: clusterrole1
rules:
  - apiGroups: ["cluster.open-cluster-management.io"]
    resources: ["managedclustersets/join"]
    resourceNames: ["clusterset1"]
    verbs: ["create"]
```

   Replace *clusterset1* with the name of your **ManagedClusterSet**.

   Note: If you are moving a managed cluster from one **ManagedClusterSet** to another, you must have that permission available on both **ManagedClusterSets**.

2. Find the definition of the managed cluster in the **yaml** file. The section of the managed cluster definition where you add a label resembles the following content:

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  name: cluster1
spec:
  hubAcceptsClient: true
```

   In this example, *cluster1* is the name of the managed cluster.

3. Add a label that specifies the name of the **ManagedClusterSet** in the format:**cluster.open-cluster-management.io/clusterset: clusterset1**.
   Your code resembles the following example:

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  name: cluster1
  labels:
    cluster.open-cluster-management.io/clusterset: clusterset1
spec:
  hubAcceptsClient: true
```

   In this example, *cluster1* is the cluster that is added to the*clusterset1* **ManagedClusterSet**.

   Note: If the managed cluster was previously assigned to a**ManagedClusterSet** that was deleted, the managed cluster might have a **ManagedClusterSet** already specified to a cluster set that does not exist. If so, replace the name with the new one.

10.3. REMOVING A MANAGED CLUSTER FROM A MANAGEDCLUSTERSET

You might want to remove a managed cluster from a **ManagedClusterSet** to move it to a different **ManagedClusterSet**, or remove it from the management settings of the set.

To remove a managed cluster from a **ManagedClusterSet**, complete the following steps:

1. Run the following command to display a list of managed clusters in the **ManagedClusterSet**:

```
kubectl get managedclusters -l cluster.open-cluster-management.io/clusterset=<clusterset1>
```

   Replace *clusterset1* with the name of the**ManagedClusterSet**.

2. Locate the entry for the cluster that you want to remove.

3. Remove the label from the the **yaml** entry for the cluster that you want to remove. See the following code for an example of the label:

```
labels:
   cluster.open-cluster-management.io/clusterset: clusterset1
```

Note: If you are moving a managed cluster from one **ManagedClusterSet** to another, you must have the RBAC permission available on both **ManagedClusterSets**.

10.4. MANAGEDCLUSTERSETBINDING RESOURCE

Create a ManagedClusterSetBinding resource to bind a ManagedClusterSet resource to a namespace. Application and policies that are created in the same namespace can only access managed clusters that are included in the bound ManagedClusterSet resource.

When you create a ManagedClusterSetBinding, the name of the ManagedClusterSetBinding must match the name of the ManagedClusterSet to bind.

Your ManagedClusterSetBinding resource might resemble the following information:

```
apiVersion: cluster.open-cluster-management.io/v1alpha1
kind: ManagedClusterSetBinding
metadata:
  namespace: project1
  name: clusterset1
spec:
  clusterSet: clusterset1
```

You must have the bind permission on the target ManagedClusterSet. View the following example of a ClusterRole resource, which contain rules that allow the user to bind to **clusterset1**:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: clusterrole1
rules:
  - apiGroups: ["cluster.open-cluster-management.io"]
    resources: ["managedclustersets/bind"]
    resourceNames: ["clusterset1"]
    verbs: ["create"]
```

For more information about role actions, see Role-based access control.

CHAPTER 11. CREATING AN ANSIBLEJOB FOR A MANAGED CLUSTER

You can create an **AnsibleJob** that is bound to a cluster when the cluster is created by Red Hat Advanced Cluster Management for Kubernetes or imported to be managed by Red Hat Advanced Cluster Management.

Complete the following procedure to create an Ansible job and configure it with a cluster that is not yet managed by Red Hat Advanced Cluster Management:

1. Create the definition file for the Ansible job in one of the channels that are supported by the application function. Only Git channels are supported.
   Use **AnsibleJob** as the value of**kind** in the definition.

   Your definition file contents might resemble the following example:

   ```
   apiVersion: apiVersion: tower.ansible.com/v1alpha1
   kind: AnsibleJob
   metadata:
     name: hive-cluster-gitrepo
   spec:
     tower_auth_secret: my-toweraccess
     job_template_name: my-tower-template-name
     extra_vars:
       variable1: value1
       variable2: value2
   ```

   By storing the file in the prehook or posthook directory, it creates a list of cluster names that match the placement rule. The list of cluster names can be passed as a value of **extra_vars** to the **AnsibleJob kind** resource. When this value is passed to the**AnsibleJob** resource, the Ansible job can determine the new cluster name and use it in the automation.

2. Log on to your Red Hat Advanced Cluster Management hub cluster.

3. Using the Red Hat Advanced Cluster Management console, create an application with a Git subscription that references the channel where you stored the definition file that you just created. See Managing application resources for more information about creating an application and subscription.
   When you create the subscription, specify a label that you can add to the cluster that you create or import later to connect this subscription with the cluster. This can be an existing label, like **vendor=OpenShift**, or a unique label that you create and define.

   Note: If you select any label that is already in use, the Ansible job automatically runs. It is recommended that you include a resource in your application that is not part of the prehooks or posthooks.

   The default placement rule runs the job when it detects the cluster with the label that matches the label of the **AnsibleJob**. If you want the automation to run on all of your running clusters that are managed by the hub cluster, add the following content to the placement rule:

   ```
   clusterConditions:
     - type: ManagedClusterConditionAvailable
       status: "True"
   ```

You can either paste this into the YAML content of the placement rule or you can select the option to *Deploy to all online clusters and local cluster* on the *Application create* page of the Red Hat Advanced Cluster Management console.

4. Create or import your cluster by following the instructions in Creating a cluster or Importing a target managed cluster to the hub cluster, respectively.
   When you create or import the cluster, use the same label that you used when you created the subscription, and the **AnsibleJob** is automatically configured to run on the cluster.

Red Hat Advanced Cluster Management automatically injects the cluster name into the **AnsibleJob.extra_vars.target_clusters** path. You can dynamically inject the cluster name into the definition. Complete the following procedure to create an AnsibleJob and configure it with a cluster that is already managed by Red Hat Advanced Cluster Management:

1. Create the definition file for the AnsibleJob in the prehook or posthook directory of your Git Channel.
   Use **AnsibleJob** as the value of **kind** in the definition.

   Your definition file contents might resemble the following example:

```
apiVersion: tower.ansible.com/v1alpha1
kind: AnsibleJob
metadata:
  name: hive-cluster-gitrepo
spec:
  tower_auth_secret: my-toweraccess
  job_template_name: my-tower-template-name
  extra_vars:
    variable1: value1
    variable2: value2
```

   Replace ***my-toweraccess*** with the authentication secret to access your Ansible Tower. Replace ***my-tower-template-name*** with the template name from your Ansible Tower.

Each time a cluster that is controlled by the Ansible job is removed or added, the AnsibleJob automatically runs and updates the **extra_vars.target_clusters** variable. This updating provides the ability to specify cluster names with a specific automation, or apply the automation to a group of clusters.

CHAPTER 12. CLUSTERCLAIMS

A ***ClusterClaim*** is a cluster-scoped custom resource definition (CRD) on a managed cluster. A ClusterClaim represents a piece of information that a managed cluster claims. The following example shows a claim that is identified in the YAML file:

```
apiVersion: cluster.open-cluster-management.io/v1alpha1
kind: ClusterClaim
metadata:
  name: id.openshift.io
spec:
  value: 95f91f25-d7a2-4fc3-9237-2ef633d8451c
```

The following table shows the defined ClusterClaims that might be on a cluster that Red Hat Advanced Cluster Management for Kubernetes manages:

Claim name	Reserved	Mutable	Description
id.k8s.io	true	false	ClusterID defined in upstream proposal
kubeversion.open-cluster-management.io	true	true	Kubernetes version
platform.open-cluster-management.io	true	false	Platform the managed cluster is running on, like AWS, GCE, and Equinix Metal
product.open-cluster-management.io	true	false	Product name, like OpenShift, Anthos, EKS and GKE
id.openshift.io	false	false	OpenShift Container Platform external ID, which is only available for an OpenShift Container Platform cluster
consoleurl.openshift.io	false	true	URL of the management console, which is only available for an OpenShift Container Platform cluster
version.openshift.io	false	true	OpenShift Container Platform version, which is only available for an OpenShift Container Platform cluster

If any of the previous claims are deleted or updated on managed cluster, they are restored or rolled back to a previous version automatically.

After the managed cluster joins the hub, the ClusterClaims that are created on a managed cluster are synchronized with the status of the **ManagedCluster** resource on the hub. A managed cluster with ClusterClaims might look similar to the following example:

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
  labels:
    cloud: Amazon
    clusterID: 95f91f25-d7a2-4fc3-9237-2ef633d8451c
    installer.name: multiclusterhub
    installer.namespace: open-cluster-management
    name: cluster1
    vendor: OpenShift
  name: cluster1
spec:
  hubAcceptsClient: true
  leaseDurationSeconds: 60
status:
  allocatable:
    cpu: '15'
    memory: 65257Mi
  capacity:
    cpu: '18'
    memory: 72001Mi
  clusterClaims:
    - name: id.k8s.io
      value: cluster1
    - name: kubeversion.open-cluster-management.io
      value: v1.18.3+6c42de8
    - name: platform.open-cluster-management.io
      value: AWS
    - name: product.open-cluster-management.io
      value: OpenShift
    - name: id.openshift.io
      value: 95f91f25-d7a2-4fc3-9237-2ef633d8451c
    - name: consoleurl.openshift.io
      value: 'https://console-openshift-console.apps.xxxx.dev04.red-chesterfield.com'
    - name: version.openshift.io
      value: '4.5'
  conditions:
    - lastTransitionTime: '2020-10-26T07:08:49Z'
      message: Accepted by hub cluster admin
      reason: HubClusterAdminAccepted
      status: 'True'
      type: HubAcceptedManagedCluster
    - lastTransitionTime: '2020-10-26T07:09:18Z'
      message: Managed cluster joined
      reason: ManagedClusterJoined
      status: 'True'
      type: ManagedClusterJoined
    - lastTransitionTime: '2020-10-30T07:20:20Z'
      message: Managed cluster is available
      reason: ManagedClusterAvailable
      status: 'True'
      type: ManagedClusterConditionAvailable
  version:
    kubernetes: v1.18.3+6c42de8
```

12.1. LIST EXISTING CLUSTERCLAIMS

You can use the **kubectl** command to list the ClusterClaims that apply to your managed cluster. This is helpful when you want to compare your ClusterClaim to an error message.

Note: Make sure you have**list** permission on resource**clusterclaims.cluster.open-cluster-management.io**.

Run the following command to list all existing ClusterClaims that are on the managed cluster:

```
kubectl get clusterclaims.cluster.open-cluster-management.io
```

12.2. CREATE CUSTOM CLUSTERCLAIMS

You can create ClusterClaims with custom names on a managed cluster, which makes it easier to identify them. The custom ClusterClaims are synchronized with the status of the **ManagedCluster** resource on the hub cluster. The following content shows an example of a definition for a customized **ClusterClaim**:

```
apiVersion: cluster.open-cluster-management.io/v1alpha1
kind: ClusterClaim
metadata:
  name: <custom_claim_name>
spec:
  value: <custom_claim_value>
```

The max length of field **spec.value** is 1024. The**create** permission on resource **clusterclaims.cluster.open-cluster-management.io** is required to create a ClusterClaim.

CHAPTER 13. SUBMARINER

The **submariner-addon** component is a technology preview feature.

Submariner is an open source tool that can be used with Red Hat Advanced Cluster Management for Kubernetes to provide direct networking between two or more Kubernetes clusters in your environment, either on-premises or in the cloud. For more information about Submariner, see Submariner.

You can enable Submariner on the OpenShift Container Platform clusters that are hosted in the following environments:

- Amazon Web Services
- Google Cloud Platform
- Microsoft Azure
- IBM Cloud
- VMware vSphere
- Bare metal
- Red Hat OpenShift Dedicated

Red Hat Advanced Cluster Management for Kubernetes provides a Submariner component that you can deploy in your environment by using your hub cluster.

13.1. PREREQUISITES

Ensure that you have the following prerequisites before using Submariner:

- A Red Hat Advanced Cluster Management hub cluster that is running on Red Hat OpenShift Container Platform version 4.5, or later, with Kubernetes version 1.17, or later.
- A credential for accessing the hub cluster with **cluster administrator** permissions.
- Two or more OpenShift Container Platform managed clusters that are running on OpenShift Container Platform version 4.4, or later, with Kubernetes version 1.17, or later, and are managed by the Red Hat Advanced Cluster Management hub cluster.
- Pod and Service Classless Inter-Domain Routing (CIDR) between the clusters that do not overlap.
- IP connectivity must be configured between the Gateway nodes. When connecting two clusters, at least one of the clusters must have a publicly routable IP address designated to the Gateway node.
- Firewall configuration across all nodes in each of the managed clusters must allow 4800/UDP in both directions.
  Note: This is configured automatically when your clusters are deployed in an Amazon Web Services environment, but must be configured manually for clusters on other environments and for the firewalls that protect private clouds like bare metal and VMware vSphere.

- Firewall configuration on the Gateway nodes allows ingress 8080/TCP so the other nodes in the cluster can access it.
- Firewall configuration open for UDP/4500, UDP/500, and any other ports that are used for IPSec traffic on the gateway nodes.

See Submariner prerequisites for more detailed information about the prerequisites.

13.2. PREPARING THE HOSTS TO DEPLOY SUBMARINER

Before deploying Submariner with Red Hat Advanced Cluster Management for Kubernetes, you must prepare the clusters on the hosting environment for the connection. The requirements vary by the hosting environment, so follow the instructions for your hosting environment.

13.2.1. Preparing Amazon Web Services to deploy Submariner

There are two ways that you can configure the OpenShift Container Platform cluster that is hosted on Amazon Web Services to integrate with a Submariner deployment. Select one of these options to prepare for the connection:

- Method 1
  You can use the **SubmarinerConfig** API to build the cluster environment. With this method, the **submariner-addon** configures the environment, so you use your configurations and cloud provider credentials in your **SubmarinerConfig** definition.

  Note: This method is only supported when the cluster is on Amazon Web Services, and not on other environments.

  Create a YAML file that contains the following content:

  ```
  apiVersion: v1
  kind: Secret
  metadata:
    name: <cloud-provider-credential-secret-name>
    namespace: <managed-cluster-namespace>
  type: Opaque
  data:
    aws_access_key_id: <aws-access-key-id>
    aws_secret_access_key: <aws-secret-access-key>
  ```

  The format of ***name*** is the same as the credential secret name that you used to provision the cluster with Red Hat Advanced Cluster Management.

  If you have already configured your Submariner cluster environment manually, include the configurations in the your **SubmarinerConfig** addition. In this example, the**IPSecIKEPort** is set to **501**, and the **IPSecNATTPort** is set to**4501**:

  ```
  apiVersion: submarineraddon.open-cluster-management.io/v1alpha1
  kind: SubmarinerConfig
  metadata:
    name: <config-name>
    namespace: <managed-cluster-namespace>
  spec:
  ```

```
IPSecIKEPort: 501
IPSecNATTPort: 4501
...
```

- Method 2
  You can use the script file, **prep_for_subm.sh**, that is provided on the Submariner website to update your OpenShift Container Platform installer-provisioned Amazon Web Services infrastructure for Submariner deployments. See Prepare AWS Clusters for Submarinerfor the script and instructions for running it.

13.2.2. Preparing Google Cloud Platform to deploy Submariner

To prepare the clusters on your Google Cloud Platform for deploying the Submariner component, complete the following steps:

1. Create the inbound and outbound firewall rules on your Google Cloud Platform to open the IPsec IKE (by default 500/UDP) and NAT traversal ports (by default 4500/UDP) to enable Submariner communication:

   ```
   $ gcloud compute firewall-rules create <name> --network=<network-name> --allow=udp:
   <ipsec-port> --direction=IN
   $ gcloud compute firewall-rules create <rule-name> --network=<network-name> --allow=udp:
   <ipsec-port>  --direction=OUT
   ```

   Replace ***rule-name*** with your rule name.

   Replace ***network-name*** with your Google Cloud Platform cluster network name.

   Replace ***ipsec-port*** with your IPsec port.

2. Create the inbound and outbound firewall rules on your Google Cloud Platform to open the 4800/UDP port to encapsulate Pod traffic from the worker and master nodes to the Submariner Gateway nodes:

   ```
   $ gcloud compute firewall-rules create <name> --network=<network-name> --
   allow=udp:4800 --direction=IN
   $ gcloud compute firewall-rules create <name> --network=<network-name> --
   allow=udp:4800 --direction=OUT
   ```

   Replace ***name*** with your rule name.

   Replace ***network-name*** with your Google Cloud Platform cluster network name.

3. Create the inbound and outbound firewall rules on your Google Cloud Platform to open the 8080/TCP port to export metrics service from the Submariner Gateway nodes:

   ```
   $ gcloud compute firewall-rules create <name> --network=<network-name> --allow=tcp:8080
   --direction=IN
   $ gcloud compute firewall-rules create <name> --network=<network-name> --allow=tcp:8080
   --direction=OUT
   ```

   Replace ***name*** with your rule name.

   Replace ***network-name*** with your Google Cloud Platform cluster network name.

13.2.3. Preparing Microsoft Azure to deploy Submariner

To prepare the clusters on your Microsoft Azure for deploying the Submariner component, complete the following steps:

1. Create the inbound and outbound firewall rules on your Microsoft Azure environment to open the IP security IKE (by default 500/UDP) and NAT traversal ports (by default 4500/UDP) to enable Submariner communication:

   ```
   # create inbound nat rule
   $ az network lb inbound-nat-rule create --lb-name <lb-name> \
   --resource-group <res-group> \
   --name <name> \
   --protocol Udp --frontend-port <ipsec-port> \
   --backend-port <ipsec-port> \
   --frontend-ip-name <frontend-ip-name>

   # add your vm network interface to the created inbound nat rule
   $ az network nic ip-config inbound-nat-rule add \
   --lb-name <lb-name> --resource-group <res-group> \
   --inbound-nat-rule <nat-name> \
   --nic-name <nic-name> --ip-config-name <pipConfig>
   ```

   Replace ***lb-name*** with your load balancer name.

   Replace ***res-group*** with your resource group name.

   Replace ***nat-name*** with your load balancing inbound NAT rule name.

   Replace ***ipsec-port*** with your IPsec port.

   Replace ***pipConfig*** with your cluster frontend IP configuration name.

   Replace ***nic-name*** with your network interface card (NIC) name.

2. Create one load balancing inbound NAT rules to forward Submariner gateway metrics service request:

   ```
   # create inbound nat rule
   $ az network lb inbound-nat-rule create --lb-name <lb-name> \
   --resource-group <res-group> \
   --name <name> \
   --protocol Tcp --frontend-port 8080 --backend-port 8080 \
   --frontend-ip-name <frontend-ip-name>

   # add your vm network interface to the created inbound nat rule
   $ az network nic ip-config inbound-nat-rule add \
   --lb-name <lb-name> --resource-group <res-group> \
   --inbound-nat-rule <nat-name> \
   --nic-name <nic-name> --ip-config-name <pipConfig>
   ```

   Replace ***lb-name*** with your load balancer name.

   Replace ***res-group*** with your resource group name.

   Replace ***nat-name*** with your load balancing inbound NAT rule name.

Replace ***pipConfig*** with your cluster frontend IP configuration name.

Replace ***nic-name*** with your network interface card (NIC) name.

3. **Create network security groups {NSG) security rules on your Azure to open IPsec IKE (by default 500/UDP) and NAT traversal ports (by default 4500/UDP) for Submariner:**

```
$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Inbound --access Allow \
--protocol Udp --destination-port-ranges <ipsec-port>

$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Outbound --access Allow \
--protocol Udp --destination-port-ranges <ipsec-port>
```

Replace ***res-group*** with your resource group name.

Replace ***nsg-name*** with your NSG name.

Replace ***priority*** with your rule priority.

Replace ***name*** with your rule name.

Replace ***ipsec-port*** with your IPsec port.

4. **Create the NSG rules to open 4800/UDP port to encapsulate pod traffic from the worker and master nodes to the Submariner Gateway nodes:**

```
$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Inbound --access Allow \
--protocol Udp --destination-port-ranges 4800 \

$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Outbound --access Allow \
--protocol Udp --destination-port-ranges 4800
```

Replace ***res-group*** with your resource group name.

Replace ***nsg-name*** with your NSG name.

Replace ***priority*** with your rule priority.

Replace ***name*** with your rule name.

5. **Create the NSG rules to open 8080/TCP port to export metrics service from the Submariner Gateway nodes:**

```
$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Inbound --access Allow \
--protocol Tcp --destination-port-ranges 8080 \
$ az network nsg rule create --resource-group <res-group> \
--nsg-name <nsg-name> --priority <priority> \
--name <name> --direction Outbound --access Allow \
--protocol Udp --destination-port-ranges 8080
```

Replace ***res-group*** with your resource group name.

Replace ***nsg-name*** with your NSG name.

Replace ***priority*** with your rule priority.

Replace ***name*** with your rule name.

13.2.4. Preparing IBM Cloud to deploy Submariner

There are two kinds of Red Hat OpenShift Kubernetes Service (ROKS) on IBM Cloud: the classic cluster and the second generation of compute infrastructure in a virtual private cloud (VPC). Submariner cannot run on the classic ROKS cluster since cannot configure the IPSec ports for the classic cluster.

To configure the ROKS clusters on a VPC to use Submariner, complete the steps in the following links:

1. Before creating a cluster, specify subnets for pods and services, which avoids overlapping CIDRs with other clusters. Make sure there are no overlapping pods and services CIDRs between clusters if you are using an existing cluster.See VPC Subnets for the procedure.

2. Attach a public gateway to subnets used in the cluster. See Public Gateway for the procedure.

3. Create inbound rules for the default security group of the cluster by completing the steps in Security Group. Ensure that the firewall allows inbound and outbound traffic on UDP/4500 and UDP/500 ports for Gateway nodes, and allows inbound and outbound UDP/4800 for all the other nodes.

4. Label a node that has the public gateway as **submariner.io/gateway=true** in the cluster.

5. Refer to Calico to configure Calico CNI by creating IPPools in the cluster.

13.2.5. Preparing Red Hat OpenShift Dedicated to deploy Submariner

Red Hat OpenShift Dedicated supports clusters that were provisioned by AWS and Google Cloud Platform.

13.2.5.1. Preparing Red Hat OpenShift Dedicated to deploy Submariner on AWS

To configure the AWS clusters on Red Hat OpenShift Dedicated, complete the following steps:

1. Submit a support ticket to the Red Hat OpenShift Hosted SRE Support team to grant **cluster-admin** group access to the Red Hat OpenShift Dedicated cluster. The default access of **dedicated-admin** does not have the permission that is required the create a **MachineSet**.

2. After the group is created, add the user name to the **cluster-admin** group that you created by completing the steps in Granting the cluster-admin role to usersin the Red Hat OpenShift Dedicated documentation.

3. Complete the prerequisites that are listed in the Preparing Amazon Web Services to deploy Submariner.

4. Configure the credentials of the user **osdCcsAdmin**, so you can use that as a service account.

13.2.5.2. Preparing Red Hat OpenShift Dedicated to deploy Submariner on Google Cloud Platform

To configure the Google Cloud Platform clusters on Red Hat OpenShift Dedicated, complete the following steps:

1. Complete the prerequisites in Preparing Google Cloud Platform to deploy Submariner.

2. Configure a service account named **osd-ccs-admin** that you can use to manage the deployment.

13.2.6. Preparing to deploy Submariner on VMware vSphere or bare metal

To prepare the VMware vSphere and bare metal clusters for deploying Submariner, complete the following steps:

1. Configure a publicly routable IP address that is designated to the gateway node on at least one of the clusters.

2. Ensure that ports for IP security are open. The default ports are 4500/UDP and 500/UDP. If the default ports are blocked by a firewall, configure a pair of custom ports that are available, like 4501/UDP and 501/UDP.

13.3. DEPLOYING SUBMARINER

The **submariner-addon** component is a technology preview feature.

Complete the following steps to deploy Submariner:

1. Log on to your hub cluster with cluster administrator permissions.

2. Ensure that you have completed the applicable preparations. See Submariner for the requirements.

3. Create a **ManagedClusterSet** on the hub cluster by using the instructions provided in ManagedClusterSets. The **submariner-addon** creates a namespace called **submariner-clusterset-<clusterset-name>-broker** and deploys the Submariner Broker to it. The name of the ManagedClusterSet replaces <clusterset-name> in the namespace name. Your entry for the **ManagedClusterSet** should resemble the following content:

```
apiVersion: cluster.open-cluster-management.io/v1alpha1
kind: ManagedClusterSet
metadata:
  name: <ManagedClusterSet-name>
```

Replace *ManagedClusterSet-name* with a name for the **ManagedClusterSet** that you are creating.

4. Enable Submariner to provide communication between managed clusters by entering the following command:

   ```
   oc label managedclusters <managedcluster-name> "cluster.open-cluster-
   management.io/submariner-agent=true" --overwrite
   ```

   Replace ***managedcluster-name*** with the name of the managed cluster that you want to use with Submariner.

5. Add the managed clusters to the **ManagedClusterSet** by entering the following command:

   ```
   oc label managedclusters <managedcluster-name> "cluster.open-cluster-
   management.io/clusterset=<ManagedClusterSet-name>" --overwrite
   ```

   Replace ***managedcluster-name*** with the name of the managed cluster that you want to add to the **ManagedClusterSet**. Replace ***ManagedClusterSet-name*** with the name of the **ManagedClusterSet** to which you want to add the managed cluster.

6. Repeat those steps for all of the managed clusters that you want to add to the **ManagedClusterSet**.

13.3.1. Deploying Submariner to an AWS OpenShift Container Platform cluster

Red Hat Advanced Cluster Management for Kubernetes version 2.2 supports automatic deployment of Submariner to managed OpenShift Container Platform cluster that are deployed on Amazon Web Services.

Complete the following steps to deploy Submariner:

1. Log on to your hub cluster with cluster administrator permissions.

   - For a cluster that was created with Red Hat Advanced Cluster Management, continue with step 2.

   - For a cluster that was imported to Red Hat Advanced Cluster Management, skip to step 3.

2. For AWS clusters that were created by Red Hat Advanced Cluster Management for Kubernetes, create a file called **submarinerconfig.yaml** with the following **SubmarinerConfig** content:

   ```
   apiVersion: submarineraddon.open-cluster-management.io/v1alpha1
   kind: SubmarinerConfig
   metadata:
     name: submariner
     namespace: <your-cluster-namespace>
   spec:
     credentialsSecret:
       name: <your-cluster-name>-aws-creds
     subscriptionConfig:
       channel: alpha
       startingCSV: submariner.v0.8.1
   ```

   - Replace **your-cluster-namespace** with your cluster’s namespace.

- Replace **your-cluster-name** with the name of your cluster. The file for**your-cluster-namespace aws-cloud-credentials** is stored in the Hive cluster namespace.
  Skip to step 4.

3. For AWS clusters that were created by OpenShift Container Platform but imported into Red Hat Advanced Cluster Management, create the **aws-cloud-credentials**.

   a. Apply the following secret to the managed cluster namespace.

   ```
   apiVersion: v1
   kind: Secret
   metadata:
     name: aws-cloud-credentials
     namespace: <your-cluster-namespace>
   type: Opaque
   data:
     aws_access_key_id: <your-aws_access_key_id>
     aws_secret_access_key: <your-aws_secret_access_key>
   ```

   b. Create a file called **submarinerconfig.yaml** with the following**SubmarinerConfig** content:

   ```
   apiVersion: submarineraddon.open-cluster-management.io/v1alpha1
   kind: SubmarinerConfig
   metadata:
     name: subconfig
     namespace: <your-cluster-namespace>
   spec:
     credentialsSecret:
       name: <aws-cloud-credentials>
     subscriptionConfig:
       channel: alpha
       startingCSV: submariner.v0.8.1
   ```

   Replace **aws-cloud-credential** with the AWS credential that you created in the previous step.

4. Apply the **SubmarinerConfig** resource to the managed cluster namespace on the hub cluster by entering the following command:

   ```
   oc apply -f submarinerconfig.yaml
   ```

5. Create a ManagedClusterSet on the hub cluster.

   ```
   apiVersion: cluster.open-cluster-management.io/v1alpha1
   kind: ManagedClusterSet
   metadata:
     name: my-clusterset
   ```

6. Add the managed clusters to the **ManagedClusterSet** by entering the following command:

   ```
   oc label managedclusters <managedcluster-name> "cluster.open-cluster-
   management.io/clusterset=<ManagedClusterSet-name>" --overwrite
   ```

7. Run the following command to allow Red Hat Advanced Cluster Management to automatically deploy Submariner:app-name:

```
oc label managedclusters <managedcluster-name> "cluster.open-cluster-
management.io/submariner-agent=true" --overwrite
```

13.4. ENABLING SERVICE DISCOVERY FOR SUBMARINER

The **submariner-addon** component is a technology preview feature.

After Submariner is deployed into the same environment as your managed clusters, the routes are configured for secure IP routing between the pod and services across the clusters in the **ManagedClusterSet**. To make a service from a cluster visible and discoverable to other clusters in the **ManagedClusterSet**, you must create a **ServiceExport** object. After a service is exported with a **ServiceExport** object, you can access the the service by the following format **<service>.<namespace>.svc.clusterset.local**. If multiple clusters export a service with the same name, and from the same namespace, they are recognized by other clusters as a single logical service.

This example uses the **nginx** service in the **default** namespace, but you can discover any Kubernetes **ClusterIP** service or headless service:

1. Apply an instance of the **nginx** service on a managed cluster that is in the **ManagedClusterSet** by entering the following commands:

```
oc -n default create deployment nginx --image=nginxinc/nginx-unprivileged:stable-alpine
oc -n default expose deployment nginx --port=8080
```

2. Export the service by creating a **ServiceExport** entry that resembles the following content in the YAML file:

```
apiVersion: multicluster.x-k8s.io/v1alpha1
kind: ServiceExport
metadata:
  name: <service-name>
  namespace: <service-namespace>
```

Replace ***service-name*** with the name of the service that you are exporting. In this example, it is **nginx**. Replace ***service-namespace*** with the name of the namespace where the service is located. In this example, it is **default**.

3. Run the following command from a different managed cluster to confirm that it can access the **nginx** service:

```
oc -n default run --generator=run-pod/v1 tmp-shell --rm -i --tty --image
quay.io/submariner/nettest -- /bin/bash curl nginx.default.svc.clusterset.local:8080
```

The **nginx** service discovery is now configured for Submariner.

CHAPTER 14. UPGRADING YOUR CLUSTER

After you create Red Hat OpenShift Container Platform clusters that you want to manage with Red Hat Advanced Cluster Management for Kubernetes, you can use the Red Hat Advanced Cluster Management for Kubernetes console to upgrade those clusters to the latest minor version that is available in the version channel that the managed cluster uses.

In a connected environment, the updates are automatically identified with notifications provided for each cluster that requires an upgrade in the Red Hat Advanced Cluster Management console.

The process for upgrading your clusters in a disconnected environment requires some additional steps to configure and mirror the required release images. It uses the operator for Red Hat OpenShift Update Service to identify the upgrades. If you are in a disconnected environment, see Upgrading disconnected clusters for the required steps.

Note: To upgrade to a major version, you must verify that you meet all of the prerequisites for upgrading to that version. You must update the version channel on the managed cluster before you can upgrade the cluster with the console. After you update the version channel on the managed cluster, the Red Hat Advanced Cluster Management for Kubernetes console displays the latest versions that are available for the upgrade.

Important: You cannot upgrade Red Hat OpenShift Kubernetes Service managed clusters or OpenShift Container Platform managed clusters on Red Hat OpenShift Dedicated by using the Red Hat Advanced Cluster Management for Kubernetes console.

This method of upgrading only works for OpenShift Container Platform managed clusters that are in a *Ready* state.

To upgrade your cluster in a connected environment, complete the following steps:

1. From the navigation menu, navigate to Automate infrastructure >Clusters. If an upgrade is available, it is shown in the *Distribution version* column.
2. Select the clusters that you want to upgrade. Remember: A cluster must be in*Ready* state, and must be a Red Hat OpenShift Container Platform cluster to be upgraded with the console.
3. Select Upgrade.
4. Select the new version of each cluster.
5. Select Upgrade.

If your cluster upgrade fails, the Operator generally retries the upgrade a few times, stops, and reports the status of the failing component. In some cases, the upgrade process continues to cycle through attempts to complete the process. Rolling your cluster back to a previous version following a failed upgrade is not supported. Contact Red Hat support for assistance if your cluster upgrade fails.

14.1. UPGRADING DISCONNECTED CLUSTERS

You can use Red Hat OpenShift Update Service with Red Hat Advanced Cluster Management for Kubernetes to upgrade your clusters in a disconnected environment.

In some cases, security concerns prevent clusters from being connected directly to the Internet. This makes it difficult to know when upgrades are available, and how to process those upgrades. Configuring OpenShift Update Service can help.

OpenShift Update Service is a separate operator and operand that monitors the available versions of your managed clusters in a disconnected environment, and makes them available for upgrading your clusters in a disconnected environment. After OpenShift Update Service is configured, it can perform the following actions:

1. Monitor when upgrades are available for your disconnected clusters.

2. Identify which updates are mirrored to your local site for upgrading by using the graph data file.

3. Notify you that an upgrade is available for your cluster by using the Red Hat Advanced Cluster Management console.

14.1.1. Prerequisites

You must have the following prerequisites before you can use OpenShift Update Service to upgrade your disconnected clusters:

- A deployed Red Hat Advanced Cluster Management hub cluster that is running on Red Hat OpenShift Container Platform version 4.5, or later with restricted OLM configured. See Using Operator Lifecycle Manager on restricted networksfor details about how to configure restricted OLM.
  Tip: Make a note of the catalog source image when you configure restricted OLM.

- An OpenShift Container Platform cluster that is managed by the Red Hat Advanced Cluster Management hub cluster

- Access credentials to a local repository where you can mirror the cluster images. See Creating a mirror registry for installation in a restricted networkfor more information about how to create this repository.
  Note: The image for the current version of the cluster that you upgrade must always be available as one of the mirrored images. If an upgrade fails, the cluster reverts back to the version of the cluster at the time that the upgrade was attempted.

14.1.2. Prepare your disconnected mirror registry

You must mirror both the image that you want to upgrade to and the current image that you are upgrading from to your local mirror registry. Complete the following steps to mirror the images:

1. Create a script file that contains content that resembles the following example:

```
UPSTREAM_REGISTRY=quay.io
PRODUCT_REPO=openshift-release-dev
RELEASE_NAME=ocp-release
OCP_RELEASE=4.5.2-x86_64
LOCAL_REGISTRY=$(hostname):5000
LOCAL_SECRET_JSON=/path/to/pull/secret

oc adm -a ${LOCAL_SECRET_JSON} release mirror \
--
from=${UPSTREAM_REGISTRY}/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELE
ASE} \
--to=${LOCAL_REGISTRY}/ocp4 \
--to-release-image=${LOCAL_REGISTRY}/ocp4/release:${OCP_RELEASE}
```

Replace **/path/to/pull/secret** with the path to your OpenShift Container Platform pull secret.

2. Run the script to mirror the images, configure settings, and separate the release images from the release content.

Tip: You can use the output of the last line of this script when you create your **ImageContentSourcePolicy**.

14.1.3. Deploy the operator for OpenShift Update Service

To deploy the operator for OpenShift Update Service in your OpenShift Container Platform environment, complete the following steps:

1. On the hub cluster, access the OpenShift Container Platform operator hub.

2. Deploy the operator by selecting **Red Hat OpenShift Update Service Operator**. Update the default values, if necessary. The deployment of the operator creates a new project named **openshift-cincinnati**.

3. Wait for the installation of the operator to finish.
   Tip: You can check the status of the installation by entering the **oc get pods** command on your OpenShift Container Platform command line. Verify that the operator is in the **running** state.

14.1.4. Build the graph data init container

OpenShift Update Service uses graph data information to determine the available upgrades. In a connected environment, OpenShift Update Service pulls the graph data information for available upgrades directly from the Cincinnati graph data GitHub repository. Because you are configuring a disconnected environment, you must make the graph data available in a local repository by using an **init container**. Complete the following steps to create a graph data **init container**:

1. Clone the *graph data* Git repository by entering the following command:

```
git clone https://github.com/openshift/cincinnati-graph-data
```

2. Create a file that contains the information for your graph data **init**. You can find this sample Dockerfile in the **cincinnati-operator** GitHub repository. The contents of the file is shown in the following sample:

```
FROM registry.access.redhat.com/ubi8/ubi:8.1

RUN curl -L -o cincinnati-graph-data.tar.gz https://github.com/openshift/cincinnati-graph-
data/archive/master.tar.gz

RUN mkdir -p /var/lib/cincinnati/graph-data/

CMD exec /bin/bash -c "tar xvzf cincinnati-graph-data.tar.gz -C /var/lib/
cincinnati/graph-data/ --strip-components=1"
```

In this example:

- The **FROM** value is the external registry where OpenShift Update Service finds the images.
- The **RUN** commands create the directory and package the upgrade files.
- The **CMD** command copies the package file to the local repository and extracts the files for an upgrade.

3. Run the following commands to build the **graph data init container**:

```
podman build -f <path_to_Dockerfile> -t
${DISCONNECTED_REGISTRY}/cincinnati/cincinnati-graph-data-container:latest
podman push ${DISCONNECTED_REGISTRY}/cincinnati/cincinnati-graph-data-
container:latest --authfile=/path/to/pull_secret.json
```

Replace *path_to_Dockerfile* with the path to the file that you created in the previous step.

Replace *${DISCONNECTED_REGISTRY}/cincinnati/cincinnati-graph-data-container* with the path to your local graph data init container.

Replace */path/to/pull_secret* with the path to your pull secret file.

Note: You can also replace**podman** in the commands with**docker**, if you don't have**podman** installed.

14.1.5. Configure certificate for the mirrored registry

If you are using a secure external container registry to store your mirrored OpenShift Container Platform release images, OpenShift Update Service requires access to this registry to build an upgrade graph. Complete the following steps to configure your CA certificate to work with the OpenShift Update Service pod:

1. Find the OpenShift Container Platform external registry API, which is located in **image.config.openshift.io**. This is where the external registry CA certificate is stored. See Configuring additional trust stores for image registry accessin the OpenShift Container Platform documentation for more information.

2. Create a ConfigMap in the **openshift-config** namespace.

3. Add your CA certificate under the key **cincinnati-registry**. OpenShift Update Service uses this setting to locate your certificate:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: trusted-ca
data:
  cincinnati-registry: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

4. Edit the **cluster** resource in the**image.config.openshift.io** API to set the **additionalTrustedCA** field to the name of the ConfigMap that you created.

```
oc patch image.config.openshift.io cluster -p '{"spec":{"additionalTrustedCA":
{"name":"trusted-ca"}}}' --type merge
```

Replace ***trusted-ca*** with the path to your new ConfigMap.

The OpenShift Update Service Operator watches the **image.config.openshift.io** API and the ConfigMap you created in the **openshift-config** namespace for changes, then restart the deployment if the CA cert has changed.

14.1.6. Deploy the OpenShift Update Service instance

When you finish deploying the OpenShift Update Service instance on your hub cluster, this instance is located where the images for the cluster upgrades are mirrored and made available to the disconnected managed cluster. Complete the following steps to deploy the instance:

1. If you do not want to use the default namespace of the operator, which is **openshift-cincinnati**, create a namespace for your OpenShift Update Service instance:

    a. In the OpenShift Container Platform hub cluster console navigation menu, select Administration >Namespaces.

    b. Select Create Namespace.

    c. Add the name of your namespace, and any other information for your namespace.

    d. Select Create to create the namespace.

2. In the *Installed Operators* section of the OpenShift Container Platform console, selectRed Hat OpenShift Update Service Operator.

3. Select Create Instance in the menu.

4. Paste the contents from your OpenShift Update Service instance. Your YAML instance might resemble the following manifest:

    ```
    apiVersion: cincinnati.openshift.io/v1beta1
    kind: Cincinnati
    metadata:
      name: openshift-update-service-instance
      namespace: openshift-cincinnati
    spec:
      registry: <registry_host_name>:<port>
      replicas: 1
      repository: ${LOCAL_REGISTRY}/ocp4/release
      graphDataImage: '<host_name>:<port>/cincinnati-graph-data-container'
    ```

    Replace the **spec.registry** value with the path to your local disconnected registry for your images.

    Replace the **spec.graphDataImage** value with the path to your graph data init container. Tip: This is the same value that you used when you ran the**podman push** command to push your graph data init container.

5. Select Create to create the instance.

6. From the hub cluster CLI, enter the **oc get pods** command to view the status of the instance creation. It might take a while, but the process is complete when the result of the command shows that the instance and the operator are running.

14.1.7. Deploy a policy to override the default registry (optional)

Note: The steps in this section only apply if you have mirrored your releases into your mirrored registry.

OpenShift Container Platform has a default image registry value that specifies where it finds the upgrade packages. In a disconnected environment, you can create a policy to replace that value with the path to your local image registry where you mirrored your release images.

For these steps, the policy is named *ImageContentSourcePolicy*. Complete the following steps to create the policy:

1. Log in to the OpenShift Container Platform environment of your hub cluster.
2. In the OpenShift Container Platform navigation, select Administration >Custom Resource Definitions.
3. Select the *Instances* tab.
4. Select the name of the *ImageContentSourcePolicy* that you created when you set up your disconnected OLM to view the contents.
5. Select the *YAML* tab to view the content in**YAML** format.
6. Copy the entire contents of the ImageContentSourcePolicy.
7. From the Red Hat Advanced Cluster Management console, select Govern risk >Create policy.
8. Set the **YAML** switch to *On* to view the YAML version of the policy.
9. Delete all of the content in the **YAML** code.
10. Paste the following **YAML** content into the window to create a custom policy:

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-pod
  namespace: default
  annotations:
    policy.open-cluster-management.io/standards:
    policy.open-cluster-management.io/categories:
    policy.open-cluster-management.io/controls:
spec:
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-pod-sample-nginx-pod
        spec:
```

```
        object-templates:
          - complianceType: musthave
            objectDefinition:
              apiVersion: v1
              kind: Pod
              metadata:
                name: sample-nginx-pod
                namespace: default
              status:
                phase: Running
        remediationAction: inform
        severity: low
  remediationAction: enforce
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-pod
  namespace: default
placementRef:
  name: placement-policy-pod
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-pod
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-pod
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      []  # selects all clusters if not specified
```

11. Replace the content inside the **objectDefinition** section of the template with content that is similar to the following content to add the settings for your ImageContentSourcePolicy:

```
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  name: ImageContentSourcePolicy
spec:
  repositoryDigestMirrors:
  - mirrors:
    - <path-to-local-mirror>
    source: registry.redhat.io
```

- Replace ***path-to-local-mirror*** with the path to your local mirror repository.

- Tip: You can find your path to your local mirror by entering the **oc adm release mirror** command.

12. Select the box for Enforce if supported.

13. Select Create to create the policy.

14.1.8. Deploy a policy to deploy a disconnected catalog source

Push the *Catalogsource* policy to the managed cluster to change the default location from a connected location to your disconnected local registry.

1. In the Red Hat Advanced Cluster Management console, select Automate infrastructure > Clusters.

2. Find the managed cluster to receive the policy in the list of clusters.

3. Note the value of the **name** label for the managed cluster. The label format is **name=managed-cluster-name**. This value is used when pushing the policy.

4. In the Red Hat Advanced Cluster Management console menu, select Govern risk > Create policy.

5. Set the **YAML** switch to *On* to view the YAML version of the policy.

6. Delete all of the content in the **YAML** code.

7. Paste the following **YAML** content into the window to create a custom policy:

8. Paste the following **YAML** content into the window to create a custom policy:

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-pod
  namespace: default
  annotations:
    policy.open-cluster-management.io/standards:
    policy.open-cluster-management.io/categories:
    policy.open-cluster-management.io/controls:
spec:
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-pod-sample-nginx-pod
        spec:
          object-templates:
            - complianceType: musthave
              objectDefinition:
                apiVersion: v1
                kind: Pod
                metadata:
                  name: sample-nginx-pod
                namespace: default
              status:
                phase: Running
          remediationAction: inform
          severity: low
  remediationAction: enforce
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-pod
  namespace: default
placementRef:
  name: placement-policy-pod
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-pod
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-pod
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      []  # selects all clusters if not specified
```

9. **Add the following content to the policy:**

```
apiVersion: config.openshift.io/vi
kind: OperatorHub
metadata:
 name: cluster
spec:
 disableAllDefaultSources: true
```

10. **Add the following content:**

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: my-operator-catalog
  namespace: openshift-marketplace
spec:
  sourceType: grpc
  image: <registry_host_name>:<port>/olm/redhat-operators:v1
  displayName: My Operator Catalog
  publisher: grpc
```

Replace the value of *spec.image* with the path to your local restricted catalog source image.

11. In the Red Hat Advanced Cluster Management console navigation, select Automate infrastructure >Clusters to check the status of the managed cluster. When the policy is applied, the cluster status is **ready**.

14.1.9. Deploy a policy to change the managed cluster parameter

Push the *ClusterVersion* policy to the managed cluster to change the default location where it retrieves its upgrades.

1. From the managed cluster, confirm that the *ClusterVersion* upstream parameter is currently the default public OpenShift Update Service operand by entering the following command:

```
oc get clusterversion -o yaml
```

The returned content might resemble the following content:

```
apiVersion: v1
items:
- apiVersion: config.openshift.io/v1
  kind: ClusterVersion
[..]
  spec:
    channel: stable-4.4
    upstream: https://api.openshift.com/api/upgrades_info/v1/graph
```

2. From the hub cluster, identify the route URL to the OpenShift Update Service operand by entering the following command: **oc get routes**.
Tip: Note this value for later steps.

3. In the hub cluster Red Hat Advanced Cluster Management console menu, select Govern risk > Create a policy.

4. Set the **YAML** switch to *On* to view the YAML version of the policy.

5. Delete all of the content in the **YAML** code.

6. Paste the following **YAML** content into the window to create a custom policy:

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-pod
  namespace: default
  annotations:
    policy.open-cluster-management.io/standards:
    policy.open-cluster-management.io/categories:
    policy.open-cluster-management.io/controls:
spec:
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-pod-sample-nginx-pod
        spec:
          object-templates:
            - complianceType: musthave
              objectDefinition:
                apiVersion: v1
                kind: Pod
                metadata:
                  name: sample-nginx-pod
                  namespace: default
                status:
                  phase: Running
          remediationAction: inform
          severity: low
  remediationAction: enforce
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-pod
  namespace: default
placementRef:
  name: placement-policy-pod
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-pod
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-pod
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      []  # selects all clusters if not specified
```

7. **Add the following content to policy.spec in the*policy* section:**

```
apiVersion: config.openshift.io/v1
  kind: ClusterVersion
  metadata:
    name: version
  spec:
    channel: stable-4.4
    upstream: https://example-cincinnati-policy-engine-uri/api/upgrades_info/v1/graph
```

Replace the value of *spec.upstream* with the path to your hub cluster OpenShift Update Service operand.

Tip: You can complete the following steps to determine the path to the operand:

a. Run the **oc get get routes -A** command on the hub cluster.

b. Find the route to **cincinnati**. + The path to the operand is the value in the **HOST/PORT** field.

8. In the managed cluster CLI, confirm that the upstream parameter in the **ClusterVersion** is updated with the local hub cluster OpenShift Update Service URL by entering:

```
oc get clusterversion -o yaml
```

Verify that the results resemble the following content:

```
apiVersion: v1
items:
- apiVersion: config.openshift.io/v1
  kind: ClusterVersion
[..]
  spec:
    channel: stable-4.4
    upstream: https://<hub-cincinnati-uri>/api/upgrades_info/v1/graph
```

14.1.10. Viewing available upgrades

You can view a list of available upgrades for your managed cluster by completing the following steps:

1. Log in to your Red Hat Advanced Cluster Management console.
2. In the navigation menu, select Automate Infrastructure >Clusters.
3. Select a cluster that is in the *Ready* state.
4. From the Actions menu, selectUpgrade cluster.
5. Verify that the optional upgrade paths are available.
   Note: No available upgrade versions are shown if the current version is not mirrored into the local image repository.

14.1.11. Upgrading the cluster

After configuring the disconnected registry, Red Hat Advanced Cluster Management and OpenShift Update Service use the disconnected registry to determine if upgrades are available. If no available upgrades are displayed, make sure that you have the release image of the current level of the cluster and at least one later level mirrored in the local repository. If the release image for the current version of the cluster is not available, no upgrades are available.

Complete the following steps to upgrade:

1. In the Red Hat Advanced Cluster Management console, select Automate infrastructure > Clusters.
2. Find the cluster that you want to determine if there is an available upgrade.

3. If there is an upgrade available, the Distribution version column for the cluster indicates that there is an upgrade available.

4. Select the *Options* menu for the cluster, and selectUpgrade cluster.

5. Select the target version for the upgrade, and select Upgrade.

The managed cluster is updated to the selected version.

If your cluster upgrade fails, the Operator generally retries the upgrade a few times, stops, and reports the status of the failing component. In some cases, the upgrade process continues to cycle through attempts to complete the process. Rolling your cluster back to a previous version following a failed upgrade is not supported. Contact Red Hat support for assistance if your cluster upgrade fails.

CHAPTER 15. REMOVING A CLUSTER FROM MANAGEMENT

When you remove an OpenShift Container Platform cluster from management that was created with Red Hat Advanced Cluster Management for Kubernetes, you can either *detach* it or *destroy* it.

Detaching a cluster removes it from management, but does not completely delete it. You can import it again, if you want to manage it. This is only an option when the cluster is in a *Ready* state.

Destroying a cluster removes it from management and deletes the components of the cluster. This is permanent, and the cluster cannot be imported and managed again.

15.1. REMOVE A CLUSTER BY USING THE CONSOLE

1. From the navigation menu, navigate to Automate infrastructure >Clusters.
2. Select the *Option* menu for the cluster that you want to remove from management.
3. Select Destroy cluster or Detach cluster.
   Tip: You can detach or destroy multiple clusters by selecting the check boxes of the clusters that you want to detach or destroy. Then select Detach or Destroy.
4. Continue with Remove remaining resources after removing a cluster.

Note: If you attempt to detach the hub cluster, which is named **local-cluster**, be aware that the default setting of **disableHubSelfManagement** is **false**. This setting causes the hub cluster to reimport itself and manage itself when it is detached and it reconciles the **MultiClusterHub** controller. It might take hours for the hub cluster to complete the detachment process and reimport. If you want to reimport the hub cluster without waiting for the processes to finish, you can enter the following command to restart the **multiclusterhub-operator** pod and reimport faster:

```
oc delete po -n open-cluster-management `oc get pod -n open-cluster-management | grep
multiclusterhub-operator| cut -d' ' -f1`
```

You can change the value of the hub cluster to not import automatically by changing the **disableHubSelfManagement** value to **true**, as described in Installing while connected online.

15.2. REMOVE A CLUSTER BY USING THE COMMAND LINE

To detach a managed cluster by using the command line of the hub cluster, run the following command:

```
oc delete managedcluster $CLUSTER_NAME
```

Note: If you attempt to detach the hub cluster, which is named **local-cluster**, be aware that the default setting of **disableHubSelfManagement** is **false**. This setting causes the hub cluster to reimport itself and manage itself when it is detached and it reconciles the **MultiClusterHub** controller. It might take hours for the hub cluster to complete the detachment process and reimport. If you want to reimport the hub cluster without waiting for the processes to finish, you can enter the following command to restart the **multiclusterhub-operator** pod and reimport faster:

```
oc delete po -n open-cluster-management `oc get pod -n open-cluster-management | grep
multiclusterhub-operator| cut -d' ' -f1`
```

You can change the value of the hub cluster to not import automatically by changing the **disableHubSelfManagement** value to**true**, as described inInstalling while connected online.

Continue with Remove remaining resources after removing a cluster

15.3. REMOVE REMAINING RESOURCES AFTER REMOVING A CLUSTER

If there are remaining resources on the managed cluster that you removed, there are additional steps that are required to ensure that you remove all of the remaining components. Situations when these extra steps are required include the following examples:

- The managed cluster was detached before it was completely created, and components like the **klusterlet** remain on the managed cluster.
- The hub that was managing the cluster was lost or destroyed before detaching the managed cluster, and there is no way to detach the managed cluster from the hub.
- The managed cluster was not in an online state when it was detached.

Complete the following steps to detach the managed cluster:

1. Make sure you have the **oc** command line interface configured.
2. Make sure you have **KUBECONFIG** configured on your managed cluster.
   If you run **oc get ns | grep open-cluster-management-agent**, you should see two namespaces:

```
open-cluster-management-agent         Active   10m
open-cluster-management-agent-addon   Active   10m
```

3. Download the **cleanup-managed-cluster** script from the**deploy** Git repository.
4. Run the **cleanup-managed-cluster.sh** script by entering the following command:

```
./cleanup-managed-cluster.sh
```

5. Run the following command to ensure that both namespaces are removed:

```
oc get ns | grep open-cluster-management-agent
```