Red Hat Advanced Cluster Management for Kubernetes 2.7

Install

Read more about installing on connected and disconnected networks, requirements and recommendations for installation, multicluster advanced configurations, and instructions for upgrading and uninstalling.

Last Updated: 2024-04-15

Red Hat Advanced Cluster Management for Kubernetes 2.7 Install

Read more about installing on connected and disconnected networks, requirements and recommendations for installation, multicluster advanced configurations, and instructions for upgrading and uninstalling.

Legal Notice

Abstract

Read more about installing on connected and disconnected networks, requirements and recommendations for installation, multicluster advanced configurations, and instructions for upgrading and uninstalling.

Table of Contents

CHAPTER 1. INSTALLING

Before you install, review the required hardware and system configuration for each product. You can install online on Linux with a supported version of Red Hat OpenShift Container Platform.

1. You must have a supported version of OpenShift Container Platform. For example, you can use Red Hat OpenShift Service on AWS, or Red Hat OpenShift Dedicated.

2. You must install the multicluster engine operator.

FIPS notice: If you do not specify your own ciphers in **spec.ingress.sslCiphers**, then the **multiclusterhub-operator** provides a default list of ciphers. For 2.3, this list includes two ciphers that are *not* FIPS approved. If you upgrade from a version 2.3.x or earlier and want FIPS compliance, remove the following two ciphers from the **multiclusterhub** resource: **ECDHE-ECDSA-CHACHA20-POLY1305** and **ECDHE-RSA-CHACHA20-POLY1305**.

Installing Red Hat Advanced Cluster Management for Kubernetes sets up a multi-node cluster production environment. You can install Red Hat Advanced Cluster Management for Kubernetes in either standard or high-availability configurations. View the following documentation for more information about the installation procedure:

- Requirements and recommendations
- Sizing your cluster
- Performance and scalability
- Installing while connected online
- Install on disconnected networks
- MultiClusterHub advanced configuration
- Upgrading
- Upgrading in a disconnected network environment
- Upgrading disconnected clusters using policies
- Uninstalling

1.1. REQUIREMENTS AND RECOMMENDATIONS

Before you install Red Hat Advanced Cluster Management for Kubernetes, review system configuration requirements and settings:

- Supported operating systems and platforms
- Supported browsers

1.1.1. Supported operating systems and platforms

Access the *Support Matrix* from the Red Hat Advanced Cluster Management for Kubernetes product information page to learn about hub cluster and managed cluster requirements and support.

1.1.2. Supported browsers

You can access the Red Hat Advanced Cluster Management console from Mozilla Firefox, Google Chrome, Microsoft Edge, and Safari. See the following versions that are tested and supported:

Platform	Supported browsers
Microsoft Windows	Microsoft Edge - 44 or later, Mozilla Firefox - 82.0 or later, Google Chrome - Version 86.0 and later
Linux	Mozilla Firefox - 82.0 and later, Google Chrome - Version 86.0 and later
macOS	Mozilla Firefox - 82.0 and later, Google Chrome - Version 86.0 and later, Safari - 14.0 and later

For cluster sizing information, see Sizing your cluster.

1.2. PERFORMANCE AND SCALABILITY

Red Hat Advanced Cluster Management for Kubernetes is tested to determine certain scalability and performance data. The major areas that are tested are cluster scalability and search performance.

You can use this information to help you plan your environment.

Note: Data is based on the results from a lab environment at the time of testing. Your results might vary, depending on your environment, network speed, and changes to the product.

- Maximum number of managed clusters
- Search scalability
- Scaling for observability

1.2.1. Maximum number of managed clusters

The maximum number of clusters that Red Hat Advanced Cluster Management can manage varies based on several factors, including:

- Number of resources in the cluster, which depends on factors like the number of policies and applications that are deployed.
- Configuration of the hub cluster, such as how many pods are used for scaling.

The following table shows the configuration information for the clusters on the Amazon Web Services cloud platform that were used during this testing:

Node	Flavor	vCPU	RAM (GiB)	Disk type	Disk size (GiB)	Count	Region
Master	m5.2xlarge	8	32	gp2	100	3	us-east-1

Node	Flavor	vCPU	RAM (GiB)	Disk type	Disk size (GiB)	Count	Region
Worker or Infrastructure	m5.2xlarge	8	32	gp2	100	3 or 5 nodes	us-east-1

For more information about infrastructure nodes, see, Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes. Also see Creating infrastructure machine sets.

1.2.2. Search scalability

The scalability of the Search component depends on the performance of the data store. The following variables are important when analyzing the search performance:

- Physical memory
- Write throughput (Cache recovery time)
- Query run time

1.2.2.1. Physical memory

Search keeps the data in-memory to achieve fast response times. The memory required is proportional to the number of Kubernetes resources and their relationships in the cluster.

Clusters	Kubernetes resources	Relationships	Observed size (with simulated data)
1 medium	5000	9500	50 Mi
5 medium	25,000	75,000	120 Mi
15 medium	75,000	20,0000	492 Mi
30 medium	150,000	450,000	1 Gi
50 medium	250,000	750,000	2 Gi

1.2.2.2. Write throughput (cache recovery time)

Most clusters in steady state generate a small number of resource updates. The highest rate of updates happen when the data in RedisGraph is cleared, which causes the remote collectors to synchronize their full state around the same time. When the datastore is cleared, recovery times are measured for a different number of managed clusters.

Clusters	Kubernetes resources	Relationships	Average recovery time from simulation
1 medium	5000	9500	less than 2 seconds
5 medium	25,000	75,000	less than 15 seconds
15 medium	75,000	200,000	2 minutes and 40 seconds
30 medium	150,000	450,000	5-8 minutes

Remember: Times might increase for clusters that have a slow network connection to the hub. The write throughput information that is previously stated is applicable only if **persistence** is disabled.

1.2.2.3. Query run time considerations

There are some things that can affect the time that it takes to run and return results from a query. Consider the following items when planning and configuring your environment:

- Searching for a keyword is not efficient.
  If you search for **RedHat** and you manage a large number of clusters, it might take a longer time to receive search results.

- The first search takes longer than later searches because it takes additional time to gather user role-based access control rules.

- The length of time to complete a request is proportional to the number of namespaces and resources the user is authorized to access.
  Note: If you save and share a Search query with another user, returned results depend on access level for that user. For more information on role access, see Using RBAC to define and apply permissions in the OpenShift Container Platform documentation.

- The worst performance is observed for a request by a non-administrator user with access to all of the namespaces, or all of the managed clusters.

1.2.3. Scaling for observability

You need to plan your environment if you want to enable and use the observability service. The resource consumption later is for the OpenShift Container Platform project, where observability components are installed. Values that you plan to use are sums for all observability components.

Note: Data is based on the results from a lab environment at the time of testing. Your results might vary, depending on your environment, network speed, and changes to the product.

1.2.3.1. Sample observability environment

In the sample environment, hub clusters and managed clusters are located in Amazon Web Services cloud platform and have the following topology and configuration:

Node	Flavor	vCPU	RAM (GiB)	Disk type	Disk size (GiB)	Count	Region
Master node	m5.4xlarge	16	64	gp2	100	3	sa-east-1
Worker node	m5.4xlarge	16	64	gp2	100	3	sa-east-1

The observability deployment is configured for high availability environments. With a high availability environment, each Kubernetes deployment has two instances, and each StatefulSet has three instances.

During the sample test, a different number of managed clusters are simulated to push metrics, and each test lasts for 24 hours. See the following throughput:

1.2.3.2. Write throughput

Pods	Interval (minute)	Time series per min
400	1	83000

1.2.3.3. CPU usage (millicores)

CPU usage is stable during testing:

Size	CPU Usage
10 clusters	400
20 clusters	800

1.2.3.4. RSS and working set memory

View the following descriptions of the RSS and working set memory:

- **Memory usage RSS:** From the metrics **container_memory_rss** and remains stable during the test.
- **Memory usage working set:** From the metrics **container_memory_working_set_bytes**, increases along with the test.

The following results are from a 24-hour test:

Size	Memory usage RSS	Memory usage working set
10 clusters	9.84	4.93

Size	Memory usage RSS	Memory usage working set
20 clusters	13.10	8.76

1.2.3.5. Persistent volume for thanos-receive component

Important: Metrics are stored in **thanos-receive** until retention time (four days) is reached. Other components do not require as much volume as **thanos-receive** components.

Disk usage increases along with the test. Data represents disk usage after one day, so the final disk usage is multiplied by four.

See the following disk usage:

Size	Disk usage (GiB)
10 clusters	2
20 clusters	3

1.2.3.6. Network transfer

During tests, network transfer provides stability. See the sizes and network transfer values:

Size	Inbound network transfer	Outbound network transfer
10 clusters	6.55 MBs per second	5.80 MBs per second
20 clusters	13.08 MBs per second	10.9 MBs per second

1.2.3.7. Amazon Simple Storage Service (S3)

Total usage in Amazon Simple Storage Service (S3) increases. The metrics data is stored in S3 until default retention time (five days) is reached. See the following disk usages:

Size	Disk usage (GiB)
10 clusters	16.2
20 clusters	23.8

1.2.4. Sizing your cluster

Each Red Hat Advanced Cluster Management for Kubernetes cluster is unique and the following guidelines provide sample deployment sizes for you. Recommendations are classified by size and purpose. Red Hat Advanced Cluster Management applies the following dimensions for sizing and placement of supporting services:

- *Availability zones* isolate potential fault domains across the cluster. Typical clusters have near equivalent worker node capacity in three or more availability zones.
- *vCPU reservations and limits* establish vCPU capacity on a worker node to assign to a container. A vCPU is equivalent to a Kubernetes compute unit. For more information, see Kubernetes Meaning of CPU.
- *Memory reservations and limits* establish memory capacity on a worker node to assign to a container.
- *Persistent data* is managed by the product and stored in the etcd cluster that is used by Kubernetes.

Important: For OpenShift Container Platform, distribute the master nodes of the cluster across three availability zones.

1.2.4.1. Product environment

Note: The following requirements are *not* minimum requirements.

Table 1.1. Product environment

Node type	Availability zones	etcd	Total reserved memory	Total reserved CPU
Master	3	3	Per OpenShift Container Platform sizing guidelines	Per OpenShift Container Platform sizing guidelines
Worker or infrastructure	3	1	12 GB	6

In addition to Red Hat Advanced Cluster Management, the OpenShift Container Platform cluster runs additional services to support cluster features. See Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes for more details.

1.2.4.1.1. OpenShift Container Platform on additional services

Availability zones isolate potential fault domains across the cluster.

Table 1.2. Additional services

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
OpenShift Container Platform on Amazon Web Services	3	3	m5.xlarge	4	16 GB	120 GB	See the Amazon Web Services information in the OpenShift Container Platform product documentation for more information. Also learn more about machine types.
OpenShift Container Platform on Google Cloud Platform	3	3	N1-standard-4 (0.95–6.5 GB)	4	15 GB	120 GB	See the Google Cloud Platform product documentation for more information about quotas. Also learn more about machine types.
OpenShift Container Platform on Microsoft Azure	3	3	Standard_D4_v3	4	16 GB	120 GB	See the following product documentation for more details.

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
OpenShift Container Platform on VMware vSphere	3	3		4 (2 cores per socket)	16 GB	120 GB	See the following product documentation for more details.
OpenShift Container Platform on IBM Z systems	3	3		10	16 GB	100 GB	See Installing a cluster on IBM Z systems in the OpenShift Container Platform documentation for more information. IBM Z systems provide the ability to configure simultaneous multithreading (SMT), which extends the number of vCPUs that can run on each core. If you configured SMT, One physical core (IFL) provides two logical cores

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
							(threads). The hypervisor can provide two or more vCPUs. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or hyperthreading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs. For more information about SMT, see Simultaneous multithreading.
OpenShift Container Platform on IBM Power systems	3	3		16	16 GB	120 GB	See Installing a cluster on Power systems in the OpenShift Container

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
							Platform documentation for more information. IBM Power systems provide the ability to configure simultaneous multithreading (SMT), which extends the number of vCPUs that can run on each core. If you configured SMT, your SMT level determines how you satisfy the 16 vCPU requirement. The most common configurations are: Two cores running on SMT-8 (the default configuration for systems that are running IBM PowerVM) provides

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
							Four cores running on SMT-4 provides the required 16 vCPUs. For more information about SMT, see Simultaneous multithreading.

Service	Node count	Availability zones	Instance size	vCPU	Memory	Storage size	Resources
OpenShift Container Platform on-premises	3			4	16 GB	120 GB	See the following product documentation for more details. A Red Hat Advanced Cluster Management for Kubernetes hub cluster can be installed and supported on OpenShift Container Platform bare metal. The hub cluster can run on a compact bare metal topology, in which there are 3 schedulable control plane nodes, and 0 additional workers.

1.2.4.1.2. Creating and managing single node OpenShift Container Platform clusters

View Installing on a single node to learn about the requirements. Since each cluster is unique, the following guidelines provide only sample deployment requirements that are classified by size and purpose.

Availability zones isolate potential fault domains across the cluster. Typical clusters have an equivalent worker node capacity in three or more availability zones. High availability is not supported.

Important: For OpenShift Container Platform, distribute the master nodes of the cluster across three availability zones.

See example requirements for creating and managing 3500 single node OpenShift Container Platform clusters. See the minimum requirements for using Red Hat Advanced Cluster Management to create single node OpenShift (SNO) clusters (230 and more provisioned at the same time), and manage those SNO clusters with a hub cluster:

Table 1.3. Master (schedulable)

Node count	Memory (peak cluster usage)	Memory (single node min-max)	CPU cluster	CPU single node
3	289 GB	64 GB - 110 GB	90	44

1.3. INSTALLING WHILE CONNECTED ONLINE

Red Hat Advanced Cluster Management for Kubernetes is installed through Operator Lifecycle Manager, which manages the installation, upgrade, and removal of the components that encompass the Red Hat Advanced Cluster Management hub cluster.

Before you get started, review the Requirements and recommendations section, then see the following documentation:

Required access: Cluster administrator.

- For OpenShift Container Platform Dedicated environment required access, you must have **cluster-admin** permissions. By default **dedicated-admin** role does not have the required permissions to create namespaces in the OpenShift Container Platform Dedicated environment.
- By default, the hub cluster components are installed on worker nodes of your OpenShift Container Platform cluster without any additional configuration. You can install the hub cluster on worker nodes by using the OpenShift Container Platform OperatorHub web console interface, or by using the OpenShift Container Platform CLI.
- If you have configured your OpenShift Container Platform cluster with infrastructure nodes, you can install the hub cluster on those infrastructure nodes by using the OpenShift Container Platform CLI with additional resource parameters. See the *Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure node* section for more details.
- If you plan to import Kubernetes clusters that were not created by OpenShift Container Platform or Red Hat Advanced Cluster Management, you need to configure an image pull secret.

For information on how to configure advanced configurations, see options in the MultiClusterHub advanced configuration section of the documentation.

- Prerequisites
- Confirm your OpenShift Container Platform installation

- Installing from the OperatorHub web console interface
- Installing from the OpenShift Container Platform CLI
- Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes

1.3.1. Prerequisites

Before you install Red Hat Advanced Cluster Management, see the following requirements:

- Your Red Hat OpenShift Container Platform cluster must have access to the Red Hat Advanced Cluster Management operator in the OperatorHub catalog from the OpenShift Container Platform console.
- You need access to the catalog.redhat.com.
- OpenShift Container Platform version 4.10, or later, must be deployed in your environment, and you must be logged into with the OpenShift Container Platform CLI. OpenShift Container Platform version 4.10, or later, must be deployed in your environment, and you must be logged into with the OpenShift Container Platform CLI. See the following install documentation for OpenShift Container Platform and change to earlier versions of needed: OpenShift Container Platform version 4.10
- Your OpenShift Container Platform command line interface (CLI) must be configured to run **oc** commands. See Getting started with the CLI for information about installing and configuring the OpenShift Container Platform CLI.
- Your OpenShift Container Platform permissions must allow you to create a namespace. Without a namespace, installation will fail.
- You must have an Internet connection to access the dependencies for the operator.
- **Important:** To install in a OpenShift Container Platform Dedicated environment, see the following requirements:
  - You must have the OpenShift Container Platform Dedicated environment configured and running.
  - You must have **cluster-admin** authority to the OpenShift Container Platform Dedicated environment where you are installing the hub cluster.
  - To import, you must use the **stable-2.0** channel of the klusterlet operator for 2.7.

1.3.2. Confirm your OpenShift Container Platform installation

You must have a supported OpenShift Container Platform version, including the registry and storage services, installed and working. For more information about installing OpenShift Container Platform, see the OpenShift Container Platform documentation.

1. Verify that a Red Hat Advanced Cluster Management hub cluster is not already installed on your OpenShift Container Platform cluster. Red Hat Advanced Cluster Management allows only one single Red Hat Advanced Cluster Management hub cluster installation on each OpenShift Container Platform cluster. Continue with the following steps if there is no Red Hat Advanced Cluster Management hub cluster installed:
2. To ensure that the OpenShift Container Platform cluster is set up correctly, access the OpenShift Container Platform web console with the following command:

```
kubectl -n openshift-console get route
```

See the following example output:

```
openshift-console console console-openshift-console.apps.new-coral.purple-chesterfield.com
console   https   reencrypt/Redirect     None
```

3. Open the URL in your browser and check the result. If the console URL displays **console-openshift-console.router.default.svc.cluster.local**, set the value for **openshift_master_default_subdomain** when you install OpenShift Container Platform. See the following example of a URL: **https://console-openshift-console.apps.new-coral.purple-chesterfield.com**.

You can proceed to install Red Hat Advanced Cluster Management from the console or the CLI. Both procedures are documented.

1.3.3. Installing from the OperatorHub web console interface

Best practice: From the *Administrator* view in your OpenShift Container Platform navigation, install the OperatorHub web console interface that is provided with OpenShift Container Platform.

1. Select **Operators** > **OperatorHub** to access the list of available operators, and select *Advanced Cluster Management for Kubernetes* operator.

2. On the *Operator subscription* page, select the options for your installation:

    - Namespace information:

        - The Red Hat Advanced Cluster Management hub cluster must be installed in its own namespace, or project.

        - By default, the OperatorHub console installation process creates a namespace titled **open-cluster-management**. **Best practice:** Continue to use the **open-cluster-management** namespace if it is available.

        - If there is already a namespace named **open-cluster-management**, choose a different namespace.

    - Channel: The channel that you select corresponds to the release that you are installing. When you select the channel, it installs the identified release, and establishes that the future Errata updates within that release are obtained.

    - Approval strategy for updates: The approval strategy identifies the human interaction that is required for applying updates to the channel or release to which you subscribed.

        - Select **Automatic** to ensure any updates within that release are automatically applied.

        - Select **Manual** to receive a notification when an update is available. If you have concerns about when the updates are applied, this might be best practice for you.

    Important: To upgrade to the next minor release, you must return to the *OperatorHub* page and select a new channel for the more current release.

3. Select **Install** to apply your changes and create the operator.

4. Create the *MultiClusterHub* custom resource.

a. In the OpenShift Container Platform console navigation, select **Installed Operators** > **Advanced Cluster Management for Kubernetes**.

b. Select the **MultiClusterHub** tab.

c. Select **Create MultiClusterHub**.

d. Update the default values in the YAML file. See options in the *MultiClusterHub advanced configuration* section of the documentation.

- The following example shows the default template. Confirm that **namespace** is your project namespace. See the sample:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
```

5. Select **Create** to initialize the custom resource. It can take up to 10 minutes for the Red Hat Advanced Cluster Management hub cluster to build and start.
After the Red Hat Advanced Cluster Management hub cluster is created, the **MultiClusterHub** resource status displays *Running* from the *MultiClusterHub* tab of the Red Hat Advanced Cluster Management operator details. To gain access to the console, see the see Accessing your console topic.

1.3.4. Installing from the OpenShift Container Platform CLI

1. Create a Red Hat Advanced Cluster Management hub cluster namespace where the operator requirements are contained. Run the following command, where **namespace** is the name for your Red Hat Advanced Cluster Management hub cluster namespace. The value for **namespace** might be referred to as *Project* in the OpenShift Container Platform environment:

```
oc create namespace <namespace>
```

2. Switch your project namespace to the one that you created. Replace **namespace** with the name of the Red Hat Advanced Cluster Management hub cluster namespace that you created in step 1.

```
oc project <namespace>
```

3. Create a YAML file to configure an **OperatorGroup** resource. Each namespace can have only one operator group. Replace **default** with the name of your operator group. Replace **namespace** with the name of your project namespace. See the following sample:

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: <default>
  namespace: <namespace>
spec:
  targetNamespaces:
  - <namespace>
```

4. Run the following command to create the **OperatorGroup** resource. Replace **operator-group** with the name of the operator group YAML file that you created:

   ```
   oc apply -f <path-to-file>/<operator-group>.yaml
   ```

5. Create a YAML file to configure an OpenShift Container Platform subscription. Your file is similar to the following sample:

   ```
   apiVersion: operators.coreos.com/v1alpha1
   kind: Subscription
   metadata:
     name: acm-operator-subscription
   spec:
     sourceNamespace: openshift-marketplace
     source: redhat-operators
     channel: release-2.5
     installPlanApproval: Automatic
     name: advanced-cluster-management
   ```

   Note: For installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes, the see the Operator Lifecycle Manager Subscription additional configuration section.

6. Run the following command to create the OpenShift Container Platform subscription. Replace **subscription** with the name of the subscription file that you created:

   ```
   oc apply -f <path-to-file>/<subscription>.yaml
   ```

7. Create a YAML file to configure the **MultiClusterHub** custom resource. Your default template should look similar to the following example. Replace **namespace** with the name of your project namespace:

   ```
   apiVersion: operator.open-cluster-management.io/v1
   kind: MultiClusterHub
   metadata:
     name: multiclusterhub
     namespace: <namespace>
   spec: {}
   ```

   Note: For installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes, see the MultiClusterHub custom resource additional configuration section:

8. Run the following command to create the **MultiClusterHub** custom resource. Replace **custom-resource** with the name of your custom resource file:

   ```
   oc apply -f <path-to-file>/<custom-resource>.yaml
   ```

   If this step fails with the following error, the resources are still being created and applied. Run the command again in a few minutes when the resources are created:

   ```
   error: unable to recognize "./mch.yaml": no matches for kind "MultiClusterHub" in version
   "operator.open-cluster-management.io/v1"
   ```

9. Run the following command to get the custom resource. It can take up to 10 minutes for the **MultiClusterHub** custom resource status to display as **Running** in the **status.phase** field after you run the command:

   ```
   oc get mch -o=jsonpath='{.items[0].status.phase}'
   ```

If you are reinstalling Red Hat Advanced Cluster Management and the pods do not start, see Troubleshooting reinstallation failure for steps to work around this problem.

Notes:

- A **ServiceAccount** with a **ClusterRoleBinding** automatically gives cluster administrator privileges to Red Hat Advanced Cluster Management and to any user credentials with access to the namespace where you install Red Hat Advanced Cluster Management.
- The installation also creates a namespace called **local-cluster** that is reserved for the Red Hat Advanced Cluster Management hub cluster when it is managed by itself. There cannot be an existing namespace called **local-cluster**. For security reasons, do not release access to the **local-cluster** namespace to any user who does not already have **cluster-administrator** access.

1.3.5. Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes

An OpenShift Container Platform cluster can be configured to contain infrastructure nodes for running approved management components. Running components on infrastructure nodes avoids allocating OpenShift Container Platform subscription quota for the nodes that are running those management components.

After adding infrastructure nodes to your OpenShift Container Platform cluster, follow the Installing from the OpenShift Container Platform CLI instructions and add configurations to the Operator Lifecycle Manager subscription and **MultiClusterHub** custom resource.

1.3.5.1. Add infrastructure nodes to the OpenShift Container Platform cluster

Follow the procedures that are described in Creating infrastructure machine sets in the OpenShift Container Platform documentation. Infrastructure nodes are configured with a Kubernetes **taint** and **label** to keep non-management workloads from running on them.

To be compatible with the infrastructure node enablement provided by Red Hat Advanced Cluster Management, ensure your infrastructure nodes have the following **taint** and **label** applied:

```
metadata:
  labels:
    node-role.kubernetes.io/infra: ""
spec:
  taints:
  - effect: NoSchedule
    key: node-role.kubernetes.io/infra
```

1.3.5.2. Operator Lifecycle Manager Subscription additional configuration

Add the following additional configuration before applying the Operator Lifecycle Manager Subscription:

```
spec:
  config:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
    tolerations:
    - key: node-role.kubernetes.io/infra
      effect: NoSchedule
      operator: Exists
```

1.3.5.3. MultiClusterHub custom resource additional configuration

Add the following additional configuration before applying the **MultiClusterHub** custom resource:

```
spec:
  nodeSelector:
    node-role.kubernetes.io/infra: ""
```

1.4. INSTALL IN DISCONNECTED NETWORK ENVIRONMENTS

You might need to install Red Hat Advanced Cluster Management for Kubernetes on disconnected Red Hat OpenShift Container Platform clusters. To install on a disconnected hub cluster, perform the following steps in addition to the usual install or upgrade steps that are for the connected network environment.

Required access: You need *cluster administration* access for all installation and upgrade tasks.

Before you get started, review the Requirements and recommendations section, then see the following sections:

- Prerequisites
- Confirm your OpenShift Container Platform installation
- Configure Operator Lifecycle Manager
- Configure image content source policies
- Install the Red Hat Advanced Cluster Management for Kubernetes operator and hub

1.4.1. Prerequisites

You must meet the following requirements before you install Red Hat Advanced Cluster Management for Kubernetes:

- Since you are installing in a disconnected network environment, you need access to a local image registry to store mirrored Operator Lifecycle Manager catalogs and operator images. You probably already set up a local image registry when installing the OpenShift Container Platform cluster in this environment, so you should be able to use the same local image registry.
- You must have a workstation that has access to both the Internet and your local mirror registry.
- A supported Red Hat OpenShift Container Platform version must be deployed in your environment, and you must be logged in with the command line interface (CLI). See the OpenShift Container Platform version 4.11 install documentation for information on installing

Red Hat OpenShift Container Platform. See Getting started with the CLI for information about installing and configuring **oc** commands with the Red Hat OpenShift CLI.

- Review Sizing your cluster to learn about setting up capacity for your hub cluster.

1.4.2. Confirm your OpenShift Container Platform installation

- While you are connected, run the **oc -n openshift-console get route** command to access the OpenShift Container Platform web console. See the following example output:

```
openshift-console        console          console-openshift-console.apps.new-coral.purple-
chesterfield.com                 console            https   reencrypt/Redirect     None
```

Open the URL in your browser and check the result. If the console URL displays **console-openshift-console.router.default.svc.cluster.local**, set the value for **openshift_master_default_subdomain** when you install OpenShift Container Platform.

1.4.3. Confirm availability of a local image registry

Best practice: Use your existing mirror registry for the Operator Lifecycle Manager operator related content.

Installing Red Hat Advanced Cluster Management for Kubernetes in a disconnected environment involves the use of a local mirror image registry. Because you have already completed the installation of the OpenShift Container Platform cluster in your disconnected environment, you already set up a mirror registry for use during the Red Hat OpenShift Container Platform cluster installation.

If you do not already have a local image registry, create one by completing the procedure that is described in Mirroring images for a disconnected installation of the Red Hat OpenShift Container Platform documentation.

1.4.4. Configure Operator Lifecycle Manager

Because Red Hat Advanced Cluster Management for Kubernetes is packaged as an operator, installing is completed by using Operator Lifecycle Manager.

In disconnected environments, Operator Lifecycle Manager cannot access the standard operator sources that Red Hat provided operators can because they are hosted on image registries that are not accessible from a disconnected cluster. Instead, a cluster administrator can enable the installation and upgrade of operators in a disconnected environment by using mirrored image registries and operator catalogs.

To prepare your disconnected cluster for installing Red Hat Advanced Cluster Management for Kubernetes, follow the procedure that is described in Using Operator Lifecycle Manager on restricted networks in the OpenShift Container Platform documentation.

1.4.4.1. Additional requirements

When you complete the previous procedures, note the following requirements that are also specific to Red Hat Advanced Cluster Management for Kubernetes:

1.4.4.1.1. Include operator packages in mirror catalog

- Include the required operator packages in your mirror catalog. Red Hat provides the Red Hat

Advanced Cluster Management for Kubernetes operator in the Red Hat operators catalog, which is delivered by the **registry.redhat.io/redhat/redhat-operator-index** index image. When you prepare your mirror of this catalog index image, you can choose to either mirror the entire catalog as provided by Red Hat, or you can mirror a subset that contains only the operator packages that you intend to use.
If you are creating a full mirror catalog, no special considerations are needed as all of the packages required to install Red Hat Advanced Cluster Management for Kubernetes are included. However, if you are creating a partial or filtered mirrored catalog, for which you identify particular packages to be included, you need to include the following package names in your list:

  - **advanced-cluster-manager**
  - **multicluster-engine**

- Use one of the two mirroring procedures.

- If you are creating the mirrored catalog or registry by using the OPM utility, **opm index prune**, include the following package names in the value of the **-p** option as displayed in the following example:

```
opm index prune \
   -f registry.redhat.io/redhat/redhat-operator-index:v4.10 \
   -p advanced-cluster-management,multicluster-engine \
   -t myregistry.example.com:5000/mirror/my-operator-index:v4.10
```

- If you are populating the mirrored catalog or registry by using the **oc-mirror** plug-in instead, include the following package names in the packages list portion of your **ImageSetConfiguration**, as displayed in the following example:

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  registry:
     imageURL: myregistry.example.com:5000/mirror/oc-mirror-metadata
mirror:
  platform:
    channels:
    - name: stable-4.10
      type: ocp
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.11
    packages:
    - name: advanced-cluster-management
    - name: multicluster-engine
  additionalImages: []
  helm: {}
```

1.4.4.1.2. Configure to use your mirror registry

When you have populated a local mirror registry with the previous packages that are required for installing Red Hat Advanced Cluster Management for Kubernetes, complete the steps that are described in the topic Using Operator Lifecycle Manager on restricted networks to make your mirror registry and catalog available on your disconnected cluster, which includes the following steps:

1. Disabling the default OperatorHub sources

2. Mirroring the Operator catalog

3. Adding a catalog source for your mirrored catalog

1.4.4.1.3. Find the catalog source name

As described in the procedures in the Red Hat OpenShift Container Platform documentation, you need to add a **CatalogSource** resource to your disconnected cluster. **Important:** Take note of the value of the **metadata.name** field, which you will need later.

Add the **CatalogSource** resource into the **openshift-marketplace** namespace by using a YAML file similar to the following example:

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: my-mirror-catalog-source
  namespace: openshift-marketplace
spec:
  image: myregistry.example.com:5000/mirror/my-operator-index:v4.10
  sourceType: grpc
```

You need the **metadata.name** field value for the annotation in the **MulticlusterHub** resource that you will create later.

1.4.5. Verify required packages are available

Operator Lifecycle Manager polls catalog sources for available packages on a regular timed interval. After Operator Lifecycle Manager polls the catalog source for your mirrored catalog, you can verify that the required packages are available from on your disconnected cluster by querying the available **PackageManifest** resources.

Run the following command, directed at your disconnected cluster:

```
oc -n openshift-marketplace get packagemanifests
```

The list that is displayed should include entries showing that the following packages are supplied by the catalog source for your mirror catalog:

- **advanced-cluster-manager**
- **multicluster-engine**

1.4.6. Configure image content source policies

In order to have your cluster obtain container images for the Red Hat Advanced Cluster Management for Kubernetes operator from your mirror registry, rather than from the internet-hosted registries, you must configure an **ImageContentSourcePolicy** on your disconnected cluster to redirect image references to your mirror registry.

If you mirrored your catalog using the **oc adm catalog mirror** command, the needed image content source policy configuration is in the **imageContentSourcePolicy.yaml** file inside of the **manifests-*** directory that is created by that command.

If you used the oc-mirror plug-in to mirror your catalog instead, the **imageContentSourcePolicy.yaml** file is within the **oc-mirror-workspace/results-*** directory create by the oc-mirror plug-in.

In either case, you can apply the policies to your disconnected command using an **oc apply** or **oc replace** command such as:

```
oc replace -f ./<path>/imageContentSourcePolicy.yaml
```

The required image content source policy statements can vary based on how you created your mirror registry, but are similar to this example:

```
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  labels:
    operators.openshift.org/catalog: "true"
  name: operator-0
spec:
  repositoryDigestMirrors:
  - mirrors:
    - myregistry.example.com:5000/rhacm2
    source: registry.redhat.io/rhacm2
  - mirrors:
    - myregistry.example.com:5000/multicluster-engine
    source: registry.redhat.io/multicluster-engine
  - mirrors:
    - myregistry.example.com:5000/openshift4
    source: registry.redhat.io/openshift4
  - mirrors:
    - myregistry.example.com:5000/redhat
    source: registry.redhat.io/redhat
```

1.4.7. Install the Red Hat Advanced Cluster Management for Kubernetes operator and hub cluster

After you have configured Operator Lifecycle Manager and Red Hat OpenShift Container Platform as previously described, you can install Red Hat Advanced Cluster Management for Kubernetes by using either the OperatorHub console or a CLI. Follow the same guidance described in the Installing while connected online topic.

Important: Creating the **MulticlusterHub** resource is the beginning of the installation process of your hub cluster.

Because operator installation on a cluster requires the use of a non-default catalog source for the mirror catalog, a special annotation is needed in the **MulticlusterHub** resource to provide the name of the mirror catalog source to the operator. The following example displays the required **mce-subscription-spec** annotation:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
   namespace: open-cluster-management
   name: hub
   annotations:
    installer.open-cluster-management.io/mce-subscription-spec: '{"source": "my-mirror-catalog-source"}'
spec: {}
```

The **mce-subscription-spec** annotation is required because multicluster engine operator is automatically installed during the Red Hat Advanced Cluster Management installation. If you are creating the resource with a CLI, include the **mce-subscription-spec** annotation in the YAML that you apply with the **oc apply** command to create the **MulticlusterHub** resource.

If you create the resource by using the OperatorHub console, switch to the *YAML view* and insert the annotation as previously displayed. **Important:** There is no field in the OperatorHub console for the annotation in the *Field view* panel to create the **MulticlusterHub**.

1.5. MULTICLUSTERHUB ADVANCED CONFIGURATION

Red Hat Advanced Cluster Management for Kubernetes is installed by using an operator that deploys all of the required components. Red Hat Advanced Cluster Management can be further configured during or after installation by adding one or more of the following attributes to the **MultiClusterHub** custom resource:

1.5.1. Console configuration

The following example displays the **spec.overrides** default template that you can use to enable or disable the **console** components. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  overrides:
    components:
    - name: console
      enabled: true
```

Alternatively, you can run the following command. Replace **namespace** with the name of your project and the value:

```
oc patch MultiClusterHub multiclusterhub -n <namespace> --type=json -p='[{"op": "add", "path": "/spec/overrides/components/-","value":{"name":"console","enabled":true}}]'
```

Note: The console is forced to stay off if the Red Hat OpenShift Container Platform console is disabled.

1.5.2. Custom Image Pull Secret

If you plan to import Kubernetes clusters that were not created by OpenShift Container Platform or Red Hat Advanced Cluster Management, generate a secret that contains your OpenShift Container Platform pull secret information to access the entitled content from the distribution registry.

The secret requirements for OpenShift Container Platform clusters are automatically resolved by OpenShift Container Platform and Red Hat Advanced Cluster Management, so you do not have to create the secret if you are not importing other types of Kubernetes clusters to be managed. Your

OpenShift Container Platform pull secret is associated with your Red Hat Customer Portal ID, and is the same across all Kubernetes providers.

Important: These secrets are namespace-specific, so make sure that you are in the namespace that you use for your hub cluster.

1. Go to cloud.redhat.com/openshift/install/pull-secret to download the OpenShift Container Platform pull secret file.
2. Click **Download pull secret**
3. Run the following command to create your secret:

   ```
   oc create secret generic <secret> -n <namespace> --from-file=.dockerconfigjson=<path-to-pull-secret> --type=kubernetes.io/dockerconfigjson
   ```

   - Replace **secret** with the name of the secret that you want to create.
   - Replace **namespace** with your project namespace, as the secrets are namespace-specific.
   - Replace **path-to-pull-secret** with the path to your OpenShift Container Platform pull secret that you downloaded.

The following example displays the **spec.imagePullSecret** template to use if you want to use a custom pull secret. Replace secret with the name of your pull secret:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  imagePullSecret: <secret>
```

1.5.3. availabilityConfig

The Red Hat Advanced Cluster Management hub cluster has two availabilities: **High** and **Basic**. By default, the hub cluster has an availability of **High**, which gives hub cluster components a **replicaCount** of **2**. This provides better support in cases of failover but consumes more resources than the **Basic** availability, which gives components a **replicaCount** of **1**.

The following examples shows the **spec.availabilityConfig** template with **Basic** availability:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  availabilityConfig: "Basic"
```

1.5.4. nodeSelector

You can define a set of node selectors in the Red Hat Advanced Cluster Management hub cluster to install to specific nodes on your cluster. The following example shows **spec.nodeSelector** to assign Red Hat Advanced Cluster Management pods to nodes with the label **node-role.kubernetes.io/infra**:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  nodeSelector:
    node-role.kubernetes.io/infra: ""
```

1.5.5. tolerations

You can define a list of tolerations to allow the Red Hat Advanced Cluster Management hub cluster to tolerate specific taints defined on the cluster.

The following example shows a **spec.tolerations** that matches a **node-role.kubernetes.io/infra** taint:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  tolerations:
  - key: node-role.kubernetes.io/infra
    effect: NoSchedule
    operator: Exists
```

The previous infra-node toleration is set on pods by default without specifying any tolerations in the configuration. Customizing tolerations in the configuration replaces this default.

1.5.6. disableHubSelfManagement

By default, the Red Hat Advanced Cluster Management hub cluster is automatically imported and managed by itself. This *managed* hub cluster is named, **local-cluster**. The setting that specifies whether a hub cluster manages itself is in the **multiclusterengine** custom resource. Changing this setting in Red Hat Advanced Cluster Management automatically changes the setting in the **multiclusterengine** custom resource.

Note: On a Red Hat Advanced Cluster Management hub cluster that is managing a multicluster engine operator cluster, any previous manual configurations are replaced by this action.

If you do not want the Red Hat Advanced Cluster Management hub cluster to manage itself, you need to change the setting for **spec.disableHubSelfManagement** from **false** to **true**. If the setting is not included in the YAML file that defines the custom resource, you need to add it. The hub cluster can only be managed with this option.

Setting this option to **true** and attempting to manage the hub manually leads to unexpected behavior.

The following example shows the default template to use if you want to disable the hub cluster self-management feature. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  disableHubSelfManagement: true
```

To enable the default **local-cluster**, return the setting to **false**, or remove this setting.

1.5.7. disableUpdateClusterImageSets

If you want to ensure that you use the same release image for all of your clusters, you can create your own custom list of release images that are available when you create a cluster.

See the following instructions in Maintaining a custom list of release images when connected to manage your available release images and to set the **spec.disableUpdateClusterImageSets** attribute, which stops the custom image list from being overwritten.

The following example shows the default template that disables updates to the cluster image set. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  disableUpdateClusterImageSets: true
```

1.5.8. customCAConfigmap (Deprecated)

By default, Red Hat OpenShift Container Platform uses the Ingress Operator to create an internal CA.

The following example shows the default template used to provide a customized OpenShift Container Platform default ingress CA certificate to Red Hat Advanced Cluster Management. Replace **namespace** with the name of your project. Replace the **spec.customCAConfigmap** value with the name of your **ConfigMap**:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  customCAConfigmap: <configmap>
```

1.5.9. sslCiphers (Deprecated)

By default, the Red Hat Advanced Cluster Management hub cluster includes the full list of supported SSL ciphers.

The following example shows the default **spec.ingress.sslCiphers** template that is used to list **sslCiphers** for the management ingress. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  ingress:
    sslCiphers:
    - "ECDHE-ECDSA-AES128-GCM-SHA256"
    - "ECDHE-RSA-AES128-GCM-SHA256"
```

1.5.10. ClusterBackup

The **enableClusterBackup** field is no longer supported and is replaced by this component.

The following example shows the **spec.overrides** default template used to enable **ClusterBackup**. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  overrides:
    components:
    - name: cluster-backup
      enabled: true
```

Alternatively, you can run the following command. Replace **namespace** with the name of your project.

```
oc patch MultiClusterHub multiclusterhub -n <namespace> --type=json -p='[{"op": "add", "path":
"/spec/overrides/components/-","value":{"name":"cluster-backup","enabled":true}}]'
```

1.5.11. ManagedServiceAccount add-on (Technology Preview)

The following example shows the **spec.overrides** default template used to enable **ManagedServiceAccount**. Replace **namespace** with the name of your project:

```
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclusterhub
  namespace: <namespace>
spec:
  overrides:
    components:
    - name: managedserviceaccount-preview
      enabled: true
```

Alternatively, you can run the following command. Replace **namespace** with the name of your project.

```
oc patch MultiClusterHub multiclusterhub -n <namespace> --type=json -p='[{"op": "add", "path":
"/spec/overrides/components/-","value":{"name":"managedserviceaccount-preview","enabled":true}}]'
```

1.6. UPGRADING

You control your Red Hat Advanced Cluster Management for Kubernetes upgrades by using the operator subscription settings in the Red Hat OpenShift Container Platform console. When you initially deploy Red Hat Advanced Cluster Management by using the operator, you make the following selections:

- **Channel:** Channel corresponds to the version of the product that you are installing. The initial channel setting is often the most current channel that was available at the time of installation.
- **Approval:** Approval specifies whether approval is required for updates within the channel, or if they are done automatically.
  - If set to **Automatic**, then minor release (Errata) updates in the selected channel are deployed without administrator intervention.
  - If set to **Manual**, then each update to the minor release (Errata) within the channel requires an administrator to approve the update.

Required access: OpenShift Container Platform administrator

You also use these settings when you upgrade to the latest version of Red Hat Advanced Cluster Management by using the operator. Complete the following steps to upgrade your operator:

Important: You cannot revert back to an earlier version after upgrading to a later version in the channel selection. You must uninstall the operator and reinstall it with the earlier version to use a previous version.

1. Log in to your OpenShift Container Platform operator hub.
2. In the OpenShift Container Platform navigation, select **Operators** > **Installed operators**.
3. Select the **Red Hat Advanced Cluster Management for Kubernetes**operator.
4. Select the *Subscription* tab to edit the subscription settings.
5. Ensure that the *Upgrade Status* is labeled *Up to date*. This status indicates that the operator is at the latest level that is available in the selected channel. If the *Upgrade Status* indicates that there is an upgrade pending, complete the following steps to update it to the latest minor release that is available in the channel:
   a. Click the **Manual** setting in the *Approval* field to edit the value.
   b. Select **Automatic** to enable automatic updates.
   c. Select **Save** to commit your change.
   d. Wait for the automatic updates to be applied to the operator. The updates automatically add the required updates to the latest version in the selected channel. When all of the updated updates are complete, the *Upgrade Status* field indicates **Up to date**.
   Note: It can take up to 10 minutes for the **MultiClusterHub** custom resource to finish upgrading. You can check whether the upgrade is still in process by entering the following command:

```
   oc get mch
   ```

   While it is upgrading, the **Status** field shows **Updating**. After upgrading is complete, the **Status** field shows **Running**.

6. Now that the *Upgrade Status* is **Up to date**, click the value in the *Channel* field to edit it.

7. Select the channel for the next available feature release, but do not attempt to skip a channel. **Important:** You cannot skip channels when upgrading. For example, you cannot skip versions 2.2.z through 2.4.

8. Select **Save** to save your changes.

9. Wait for the automatic upgrade to complete. After the upgrade to the next feature release completes, the updates to the latest patch releases within the channel are deployed.

10. If you have to upgrade to a later feature release, repeat steps 7-9 until your operator is at the latest level of the desired channel. Make sure that all of the patch releases are deployed for your final channel.

11. Optional: You can set your *Approval* setting to **Manual**, if you want your future updates within the channel to require manual approvals.

For more information about upgrading your operator, see Operators in the OpenShift Container Platform documentation.

1.6.1. Managing cluster pools with an upgrade

If you are Managing cluster pools (Technology Preview), you need further configuration to stop automatic management of these cluster pools after upgrade.

Set **cluster.open-cluster-management.io/createmanagedcluster: "false"** in the **ClusterClaim metadata.annotations**.

All existing cluster claims are automatically imported when the product is upgraded unless you change this setting.

1.7. UPGRADING IN A DISCONNECTED NETWORK ENVIRONMENT

See the steps and information to upgrade Red Hat Advanced Cluster Management for Kubernetes in a disconnected network environment.

Note: This information follows the upgrading procedure in Upgrading. Review that procedure, then see the following information:

1.7.1. Upgrade from release 2.5 or later

During your installation, or upgrade, of Red Hat Advanced Cluster Management for Kubernetes to release 2.5 or later, you encountered important information that is related to the interdependency between the Red Hat Advanced Cluster Management for Kubernetes and multicluster engine for Kubernetes operators. See Install in disconnected network environments . Similar considerations are required when you upgrade.

As is the case for upgrading in a connected network environment, the upgrade process is started by changing the upgrade channel in your Operator Lifecycle Manager subscription for Red Hat Advanced Cluster Management for Kubernetes to the upgrade channel for the new release.

However, because of the special characteristics of the disconnected environment, you need to address the following mirroring requirements before changing the update channel to start the upgrade process:

1. Ensure that required packages are updated in your mirror catalog.
   During installation, or during a previous update, you created a mirror catalog and a registry that contains operator packages and images that are needed to install Red Hat Advanced Cluster Management for Kubernetes in a disconnected network environment. To upgrade, you need to update your mirror catalog and registry to pick up the updated versions of the operator packages.

   Similar to your installation actions, you need to ensure that your mirror catalog and registry include the following operator packages in the list of operators to be included or updated:

   - **advanced-cluster-manager**

   - **multicluster-engine**

2. Verify your **MutliclusterHub** resource instance.
   During installation or a previous update, you created an instance of the **MulticlusterHub** resource, and due to the disconnected environment, you added a **mce-subscription-spec** annotation to that resource.

   If your procedures for updating your mirror catalog and registry resulted in the updated catalog being available on the OpenShift Container Platform cluster through a **CatalogSource** with the same name as the one that you previously used, you do not need to update your **MulticlusterHub** resource to update the **mce-subscriptino-spec** annotation.

   However, if your procedures for updating your mirrored catalog and registry resulted in a newly named **CatalogSource** being created, update the **mce-subscription-spec** annotation in your **MulticlusterHub** resource to reflect the new catalog source name.

1.7.2. Upgrade from release 2.4

Red Hat Advanced Cluster Management for Kubernetes release 2.5 and later uses the related multicluster engine for Kubernetes operator functionality to provide foundational services that previously were delivered as part of Red Hat Advanced Cluster Management for Kubernetes. Releases 2.5 and later of the Red Hat Advanced Cluster Management for Kubernetes operator automatically install and manage the required multicluster engine for Kubernetes operator and **MulticlusterEngine** resource instance as part of the hub cluster installation and upgrade.

In connected network environments, the cluster administrator can install or upgrade Red Hat Advanced Cluster Management for Kubernetes without special mirror catalogs and catalog sources. However, because installation of any Operator Lifecycle Manager operator in a disconnected environment involves the use of special mirror catalogs and catalog sources (as described in the earlier sections), some additional steps are necessary beyond installation.

1. Update your procedures for populating the mirror catalog
   If, when installing Red Hat Advanced Cluster Management for Kubernetes release 2.4 and later, mirroring procedures created a full copy of the Red Hat Operators catalog, no special mirroring updates are required. Refresh your catalog to pick up the updated content for the new operator releases.

However, if your procedures populated mirror catalog that is a *filtered* catalog, you need to update your mirroring procedures to ensure that the **multcluster-engine** operator package is included in the mirror catalog, in addition to the **advanced-cluster-management** package.

See the Include required operator packages in your mirror catalog topic, which provides examples of the options to use when populating the mirror catalog. Update the operator-package lists that are used in your procedures to match these new requirements.

2. Update your **MutliclusterHub** resource instance.
   As described in the Install in disconnected network environments topic, you need a new annotation on the **MulticlusterHub** resource when the hub cluster is installed or upgraded in a disconnected environment.

   Best practice: Update your **MulticlusterHub** resource instance to include the required annotation before you change the Operator Lifecycle Manager update channel in your Operator Lifecycle Manager subscription to the **advanced-cluster-management** operator package to start the upgrade from release 2.4. This update allows the upgrade to proceed without delay.

   Use the **oc edit** command to update your **Multiclusterub** resource to add the **mce-subscription-spec** annotation as displayed in the following example:

   ```
   metadata:
     annotations:
       installer.open-cluster-management.io/mce-subscription-spec: '{"source": "<my-mirror-catalog-source>"}'
   ```

   Replace **<my-mirror-catalog-source>** from the example with the name of the **CatalogSource** resource located in the **openshift-marketplace** namespace for your mirror catalog.

Important: If you begin an upgrade from release 2.4 to release 2.5 before you add the annotation, the upgrade begins but stalls when the operator attempts to install a subscription to **multicluster-engine** in the background. The status of the **MulticlusterHub** resource continues to display **upgrading** during this time.

To resolve this issue, run **oc edit** to add the **mce-subscription-spec** annotation as shown previously.

1.8. UPGRADING DISCONNECTED CLUSTERS USING POLICIES

You can use Red Hat OpenShift Update Service with Red Hat Advanced Cluster Management for Kubernetes policies to upgrade multiple 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 console.

1.8.1. Prerequisites

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

- A deployed hub cluster that is running on Red Hat OpenShift Container Platform version 4.6 or later with restricted OLM configured. See Using Operator Lifecycle Manager on restricted networks for 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 hub cluster

- Access credentials to a local repository where you can mirror the cluster images. See Disconnected installation mirroring for 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.

1.8.2. 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=4.12.2-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**.

1.8.3. 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.

1.8.4. 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.

1.8.5. 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 access in 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 **updateservice-registry**. OpenShift Update Service uses this setting to locate your certificate:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: trusted-ca
data:
  updateservice-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.

1.8.6. 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, select **Red 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/v1beta2
kind: Cincinnati
metadata:
  name: openshift-update-service-instance
  namespace: openshift-cincinnati
spec:
  registry: <registry_host_name>:<port> 1
  replicas: 1
  repository: ${LOCAL_REGISTRY}/ocp4/release
  graphDataImage: '<host_name>:<port>/cincinnati-graph-data-container' 2
```

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

2 Replace the **spec.graphDataImage** value with the path to your graph data init container. 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.

1.8.7. 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 *policy-mirror*. Complete the following steps to create the policy:

1. Log in to the OpenShift Container Platform environment of your hub cluster.

2. From the console, select **Governance** > **Create policy**.

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

4. Delete all of the content in the YAML code.

5. 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-mirror
  namespace: default
spec:
  disabled: false
  remediationAction: enforce
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-image-content-source-policy
        spec:
          object-templates:
            - complianceType: musthave
              objectDefinition:
                apiVersion: operator.openshift.io/v1alpha1
                kind: ImageContentSourcePolicy
                metadata:
                  name: <your-local-mirror-name>
                spec:
                  repositoryDigestMirrors:
                    - mirrors:
                        - <your-registry> 1
                      source: registry.redhat.io
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-mirror
  namespace: default
placementRef:
  name: placement-policy-mirror
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-mirror
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-mirror
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      []  # selects all clusters if not specified
```

1 Replace **your-registry** with the path to your local mirror repository. You can find your path to your local mirror by entering the **oc adm release mirror** command.

6. Select **Enforce if supported**.

7. Select **Create** to create the policy.

1.8.8. 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 console menu, select **Governance** > **Create policy**.

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

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

4. 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-catalog
  namespace: default
spec:
  disabled: false
  remediationAction: enforce
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-catalog
        spec:
          object-templates:
            - complianceType: musthave
              objectDefinition:
                apiVersion: config.openshift.io/v1
                kind: OperatorHub
                metadata:
                  name: cluster
                spec:
                  disableAllDefaultSources: true
            - complianceType: musthave
              objectDefinition:
                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' 1
                  displayName: My Operator Catalog
                  publisher: grpc
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-catalog
  namespace: default
placementRef:
  name: placement-policy-catalog
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-catalog
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-catalog
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      []  # selects all clusters if not specified
```

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

5. Select **Enforce if supported**.

6. Select **Create** to create the policy.

1.8.9. 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**. Note this value for later steps.

3. In the hub cluster console menu, select **Governance** > **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-cluster-version
  namespace: default
  annotations:
    policy.open-cluster-management.io/standards: null
    policy.open-cluster-management.io/categories: null
    policy.open-cluster-management.io/controls: null
spec:
  disabled: false
  remediationAction: enforce
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-cluster-version
        spec:
          object-templates:
            - complianceType: musthave
              objectDefinition:
                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 1

---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-cluster-version
  namespace: default
placementRef:
  name: placement-policy-cluster-version
  kind: PlacementRule
  apiGroup: apps.open-cluster-management.io
subjects:
- name: policy-cluster-version
  kind: Policy
  apiGroup: policy.open-cluster-management.io
---
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-policy-cluster-version
  namespace: default
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      [] # selects all clusters if not specified
```

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

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.

7. Select **Enforce if supported**.

8. Select **Create** to create the policy.

9. 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
```

1.8.10. 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 multicluster engine for Kubernetes operator console.
2. In the navigation menu, select **Infrastructure** > **Clusters**.
3. Select a cluster that is in the *Ready* state.
4. From the **Actions** menu, select **Upgrade 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.

1.8.11. Selecting a channel

You can use the Red Hat Advanced Cluster Management console to select a channel for your cluster upgrades on OpenShift Container Platform version 4.6, or later. Those versions must be available on the mirror registry. Complete the steps in Selecting a channel to specify a channel for your upgrades.

1.8.12. 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 console, select **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 select **Upgrade 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.

1.9. UNINSTALLING

When you uninstall Red Hat Advanced Cluster Management for Kubernetes, you see two different levels of the uninstall process: A *custom resource removal* and a *complete operator uninstall*. The uninstall process can take up to 20 minutes.

- The first level is the custom resource removal, which is the most basic type of uninstall that removes the custom resource of the **MultiClusterHub** instance, but leaves other required operator resources. This level of uninstall is helpful if you plan to reinstall using the same settings and components.

- The second level is a more complete uninstall that removes most operator components, excluding components such as custom resource definitions. When you continue with this step, it removes all of the components and subscriptions that were not removed with the custom resource removal. After this uninstall, you must reinstall the operator before reinstalling the custom resource.

1.9.1. Prerequisite: Detach enabled services

Before you uninstall the Red Hat Advanced Cluster Management hub cluster, you must detach all of the clusters that are managed by that hub cluster. To resolve errors, detach all clusters that are still managed by the hub cluster, then try to uninstall again.

- If you use Discovery, you might see the following error when you attempt uninstall:

  ```
  Cannot delete MultiClusterHub resource because DiscoveryConfig resource(s) exist
  ```

  To disable Discovery, complete the following steps:

  - From the console, navigate to the **Discovered Clusters** table and click **Disable cluster discovery**. Confirm that you want to remove the service.

  - You can also use the terminal. Run the following command to disable Discover:

  ```
  $ oc delete discoveryconfigs --all --all-namespaces
  ```

- If you have managed clusters attached, you might see the following message. **Note:** This does not include the **local-cluster**, which is your self-managed hub cluster:

  ```
  Cannot delete MultiClusterHub resource because ManagedCluster resource(s) exist
  ```

  For more information about detaching clusters, see the *Removing a cluster from management* section by selecting the information for your provider in Creating a cluster.

- If you have Observability, you might see the following:

  ```
  Cannot delete MultiClusterHub resource because MultiClusterObservability resource(s) exist
  ```

- To disable and remove the **MultiClusterObservability** using the terminal, see the following procedure:

  a. Log in to your hub cluster.

  b. Delete the **MultiClusterObservability** custom resource by entering the following command:

  ```
  oc delete mco observability
  ```

- To remove **MultiClusterObservability** custom resource with the console, see the following procedure:

  a. If the **MultiClusterObservability** custom resource is installed, select the tab for *MultiClusterObservability*.

  b. Select the *Options* menu for the **MultiClusterObservability** custom resource.

  c. Select **Delete MultiClusterObservability**.
  When you delete the resource, the pods in the **open-cluster-management-observability** namespace on Red Hat Advanced Cluster Management hub cluster, and the pods in **open-cluster-management-addon-observability** namespace on all managed clusters are removed.

Note: Your object storage is not affected after you remove the observability service.

1.9.2. Removing resources by using commands

1. If you have not already. ensure that your OpenShift Container Platform CLI is configured to run **oc** commands. See Getting started with the OpenShift CLI in the OpenShift Container Platform documentation for more information about how to configure the **oc** commands.

2. Change to your project namespace by entering the following command. Replace *namespace* with the name of your project namespace:

   ```
   oc project <namespace>
   ```

3. Enter the following command to remove the **MultiClusterHub** custom resource:

   ```
   oc delete multiclusterhub --all
   ```

   You can view the progress by entering the following command:

   ```
   oc get mch -o yaml
   ```

4. Remove any potential remaining artifacts by running the clean-up script. Run this clean-up script if you plan to reinstall with an older version of Red Hat Advanced Cluster Management on the same cluster.

   a. Install the Helm CLI binary version 3.2.0, or later, by following the instructions at Installing Helm.

   b. Copy the following script into a file:

   ```
   #!/bin/bash
   ```

```
ACM_NAMESPACE=<namespace>
oc delete mch --all -n $ACM_NAMESPACE
helm ls --namespace $ACM_NAMESPACE | cut -f 1 | tail -n +2 | xargs -n 1 helm delete -
-namespace $ACM_NAMESPACE
oc delete apiservice v1beta2.webhook.certmanager.k8s.io v1.admission.cluster.open-
cluster-management.io v1.admission.work.open-cluster-management.io
oc delete clusterimageset --all
oc delete clusterrole multiclusterengines.multicluster.openshift.io-v1-admin
multiclusterengines.multicluster.openshift.io-v1-crdview
multiclusterengines.multicluster.openshift.io-v1-edit
multiclusterengines.multicluster.openshift.io-v1-view open-cluster-
management:addons:application-manager open-cluster-management:admin-aggregate
open-cluster-management:cert-policy-controller-hub open-cluster-management:cluster-
manager-admin-aggregate open-cluster-management:config-policy-controller-hub open-
cluster-management:edit-aggregate open-cluster-management:iam-policy-controller-hub
open-cluster-management:policy-framework-hub open-cluster-management:view-
aggregate
oc delete configmap -n $ACM_NAMESPACE cert-manager-controller cert-manager-
cainjector-leader-election cert-manager-cainjector-leader-election-core
oc delete consolelink acm-console-link
oc delete crd klusterletaddonconfigs.agent.open-cluster-management.io
placementbindings.policy.open-cluster-management.io policies.policy.open-cluster-
management.io userpreferences.console.open-cluster-management.io
searchservices.search.acm.com discoveredclusters.discovery.open-cluster-
management.io discoveryconfigs.discovery.open-cluster-management.io
oc delete mutatingwebhookconfiguration cert-manager-webhook cert-manager-webhook-
v1alpha1 ocm-mutating-webhook managedclustermutators.admission.cluster.open-
cluster-management.io multicluster-observability-operator
oc delete oauthclient multicloudingress
oc delete rolebinding -n kube-system cert-manager-webhook-webhook-authentication-
reader
oc delete scc kui-proxy-scc
oc delete validatingwebhookconfiguration cert-manager-webhook cert-manager-
webhook-v1alpha1 channels.apps.open.cluster.management.webhook.validator
application-webhook-validator multiclusterhub-operator-validating-webhook ocm-
validating-webhook multicluster-observability-operator
multiclusterengines.multicluster.openshift.io
```

Replace **<namespace>** in the script with the name of the namespace where Red Hat Advanced Cluster Management was installed. Ensure that you specify the correct namespace, as the namespace is cleaned out and deleted.

c. Run the script to remove any possible artifacts that remain from the previous installation. If there are no remaining artifacts, a message is returned that no resources were found.
Note: If you plan to reinstall the same Red Hat Advanced Cluster Management version, you can skip the next steps in this procedure and reinstall the custom resource. Proceed for a complete operator uninstall.

5. Enter the following commands to delete the Red Hat Advanced Cluster Management **ClusterServiceVersion** and **Subscription** in the namespace it is installed in:

```
❯ oc get csv
NAME                                 DISPLAY                                      VERSION   REPLACES   PHASE
advanced-cluster-management.v2.x.0   Advanced Cluster Management for Kubernetes   2.x.0
Succeeded
```

```
❯ oc delete clusterserviceversion advanced-cluster-management.v2.x.0

❯ oc get sub
NAME                       PACKAGE                       SOURCE                CHANNEL
acm-operator-subscription  advanced-cluster-management   acm-custom-registry   release-2.x

❯ oc delete sub acm-operator-subscription
```

Note: The name of the subscription and version of the CSV might differ.

1.9.3. Deleting the components by using the console

When you use the Red Hat OpenShift Container Platform console to uninstall, you remove the operator. Complete the following steps to uninstall by using the console:

1. In the OpenShift Container Platform console navigation, select **Operators** > **Installed Operators** > **Advanced Cluster Manager for Kubernetes**.

2. Remove the **MultiClusterHub** custom resource.

   a. Select the tab for *Multiclusterhub*.

   b. Select the *Options* menu for the MultiClusterHub custom resource.

   c. Select **Delete MultiClusterHub**.

3. Run the clean-up script according to the procedure in Removing a MultiClusterHub instance by using commands.
   Tip: If you plan to reinstall the same Red Hat Advanced Cluster Management version, you can skip the rest of the steps in this procedure and reinstall the custom resource.

4. Navigate to **Installed Operators**.

5. Remove the *Red Hat Advanced Cluster Management* operator by selecting the *Options* menu and selecting **Uninstall operator**.