



# Red Hat OpenShift AI Self-Managed 2.10

## Installing and uninstalling OpenShift AI Self-Managed in a disconnected environment

Install and uninstall OpenShift AI Self-Managed in a disconnected environment



## Red Hat OpenShift AI Self-Managed 2.10 Installing and uninstalling OpenShift AI Self-Managed in a disconnected environment

---

Install and uninstall OpenShift AI Self-Managed in a disconnected environment

## Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

## Abstract

Install and uninstall OpenShift AI Self-Managed on your OpenShift Container Platform cluster in a disconnected environment.

# Table of Contents

<b>PREFACE</b> .....	<b>3</b>
<b>CHAPTER 1. ARCHITECTURE OF OPENSIFT AI SELF-MANAGED</b> .....	<b>4</b>
<b>CHAPTER 2. UNDERSTANDING UPDATE CHANNELS</b> .....	<b>6</b>
<b>CHAPTER 3. DEPLOYING OPENSIFT AI IN A DISCONNECTED ENVIRONMENT</b> .....	<b>8</b>
3.1. REQUIREMENTS FOR OPENSIFT AI SELF-MANAGED	8
3.2. ADDING ADMINISTRATIVE USERS FOR OPENSIFT CONTAINER PLATFORM	10
3.3. MIRRORING IMAGES TO A PRIVATE REGISTRY FOR A DISCONNECTED INSTALLATION	10
3.4. INSTALLING THE RED HAT OPENSIFT AI OPERATOR	14
3.4.1. Installing the Red Hat OpenShift AI Operator by using the CLI	15
3.4.2. Installing the Red Hat OpenShift AI Operator by using the web console	17
3.5. INSTALLING AND MANAGING RED HAT OPENSIFT AI COMPONENTS	19
3.5.1. Installing Red Hat OpenShift AI components by using the CLI	19
3.5.2. Installing Red Hat OpenShift AI components by using the web console	22
3.5.3. Updating the installation status of Red Hat OpenShift AI components by using the web console	23
3.5.4. Disabling KServe dependencies	26
3.6. CONFIGURING THE OPENSIFT AI OPERATOR LOGGER	26
3.6.1. Configuring the OpenShift AI Operator logger	26
3.6.1.1. Viewing the OpenShift AI Operator log	28
<b>CHAPTER 4. PREPARING OPENSIFT AI FOR USE IN IBM CLOUD PAK FOR DATA</b> .....	<b>29</b>
4.1. INSTALLING THE RED HAT OPENSIFT AI OPERATOR BY USING THE CLI	29
4.2. MANAGING RED HAT OPENSIFT AI COMPONENTS BY USING THE CLI	32
4.3. EDITING THE MODEL INFERENCING CONFIGURATION	34
<b>CHAPTER 5. WORKING WITH CERTIFICATES</b> .....	<b>36</b>
5.1. UNDERSTANDING CERTIFICATES IN OPENSIFT AI	36
5.1.1. How CA bundles are injected	36
5.1.2. How the ConfigMap is managed	36
5.2. ADDING A CA BUNDLE	37
5.3. REMOVING A CA BUNDLE	39
5.4. REMOVING A CA BUNDLE FROM A NAMESPACE	39
5.5. MANAGING CERTIFICATES	40
5.6. USING SELF-SIGNED CERTIFICATES WITH OPENSIFT AI COMPONENTS	41
5.6.1. Using certificates with data science pipelines	41
5.6.1.1. Providing a CA bundle only for data science pipelines	41
5.6.2. Using certificates with workbenches	43
5.6.2.1. Creating data science pipelines with Elyra and self-signed certificates	43
<b>CHAPTER 6. ENABLING GPU SUPPORT IN OPENSIFT AI</b> .....	<b>44</b>
<b>CHAPTER 7. ACCESSING THE DASHBOARD</b> .....	<b>46</b>
<b>CHAPTER 8. UNINSTALLING RED HAT OPENSIFT AI SELF-MANAGED</b> .....	<b>47</b>
8.1. UNDERSTANDING THE UNINSTALLATION PROCESS	47
8.2. UNINSTALLING OPENSIFT AI SELF-MANAGED BY USING THE CLI	48



---

## PREFACE

Learn how to use both the OpenShift command-line interface and web console to install Red Hat OpenShift AI Self-Managed on your OpenShift Container Platform cluster in a disconnected environment. To uninstall the product, learn how to use the recommended command-line interface (CLI) method.



### NOTE

Red Hat recommends that you install only one instance of OpenShift AI on your cluster.

Installing the Red Hat OpenShift AI Operator on the same cluster as the Red Hat OpenShift AI Add-on is not recommended or supported.

# CHAPTER 1. ARCHITECTURE OF OPENSIFT AI SELF-MANAGED

Red Hat OpenShift AI Self-Managed is an Operator that is available on a self-managed environment, such as Red Hat OpenShift Container Platform.

OpenShift AI integrates the following components and services:

- At the service layer:

## OpenShift AI dashboard

A customer-facing dashboard that shows available and installed applications for the OpenShift AI environment as well as learning resources such as tutorials, quick starts, and documentation. Administrative users can access functionality to manage users, clusters, notebook images, accelerator profiles, and model-serving runtimes. Data scientists can use the dashboard to create projects to organize their data science work.

## Model serving

Data scientists can deploy trained machine-learning models to serve intelligent applications in production. After deployment, applications can send requests to the model using its deployed API endpoint.

## Data science pipelines

Data scientists can build portable machine learning (ML) workflows with Data Science Pipelines 2.0, using Docker containers. With data science pipelines, data scientists can automate workflows as they develop their data science models.

## Jupyter (self-managed)

A self-managed application that allows data scientists to configure their own notebook server environment and develop machine learning models in JupyterLab.

## Distributed workloads

Data scientists can use multiple nodes in parallel to train machine-learning models or process data more quickly. This approach significantly reduces the task completion time, and enables the use of larger datasets and more complex models.

- At the management layer:

## The Red Hat OpenShift AI Operator

A meta-operator that deploys and maintains all components and sub-operators that are part of OpenShift AI.

## Monitoring services

Prometheus gathers metrics from OpenShift AI for monitoring purposes.

When you install the Red Hat OpenShift AI Operator in the OpenShift Container Platform cluster, the following new projects are created:

- The **redhat-ods-operator** project contains the Red Hat OpenShift AI Operator.
- The **redhat-ods-applications** project installs the dashboard and other required components of OpenShift AI.
- The **redhat-ods-monitoring** project contains services for monitoring.
- The **rhods-notebooks** project is where notebook environments are deployed by default.



You or your data scientists must create additional projects for the applications that will use your machine learning models.

Do not install independent software vendor (ISV) applications in namespaces associated with OpenShift AI.

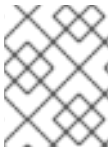
## CHAPTER 2. UNDERSTANDING UPDATE CHANNELS

You can use update channels to specify which Red Hat OpenShift AI minor version you intended to update your Operator to. Update channels also allow you to choose the timing and level of support your updates have through the **fast**, **stable**, **stable-x.y**, **eus-x.y**, and **alpha** channel options.

The subscription of an installed Operator specifies the update channel, which is used to track and receive updates for the Operator. You can change the update channel to start tracking and receiving updates from a newer channel. For more information about the release frequency and the lifecycle associated with each of the available update channels, see [Red Hat OpenShift AI Self-Managed Life Cycle](#).

Channel	Support	Release frequency	Recommended environment
<b>fast</b>	One month of full support	Every month	<p>Production environments with access to the latest product features.</p> <p>Select this streaming channel with automatic updates to avoid manually upgrading every month.</p>
<b>stable</b>	Three months of full support	Every three months	<p>Production environments with stability prioritized over new feature availability.</p> <p>Select this streaming channel with automatic updates to access the latest stable release and avoid manually upgrading.</p>
<b>stable-x.y</b>	Seven months of full support	Every three months	<p>Production environments with stability prioritized over new feature availability.</p> <p>Select numbered stable channels (such as <b>stable-2.10</b>) to plan and upgrade to the next stable release while keeping your deployment under full support.</p>
<b>eus-x.y</b>	Seven months of full support followed by Extended Update Support for eleven months	Every nine months	<p>Enterprise-grade environments that cannot upgrade within a seven month window.</p> <p>Select this streaming channel if you prioritize stability over new feature availability.</p>

Channel	Support	Release frequency	Recommended environment
<b>alpha</b>	One month of full support	Every month	<p>Development environments with early-access features that might not be functionally complete.</p> <p>Select this channel to use early-access features to test functionality and provide feedback during the development process. Early-access features are not supported with Red Hat production service level agreements (SLAs).</p> <p>For more information about the support scope of Red Hat Technology Preview features, see <a href="#">Technology Preview Features Support Scope</a>.</p> <p>For more information about the support scope of Red Hat Developer Preview features, see <a href="#">Developer Preview Features Support Scope</a>.</p>



#### NOTE

The **embedded** and **beta** channels are legacy channels that will be removed in a future release. Do not select the **embedded** or **beta** channels for a new Operator installation.

## CHAPTER 3. DEPLOYING OPENSIFT AI IN A DISCONNECTED ENVIRONMENT

Read this section to understand how to deploy Red Hat OpenShift AI as a development and testing environment for data scientists in a disconnected environment. Disconnected clusters are on a restricted network, typically behind a firewall. In this case, clusters cannot access the remote registries where Red Hat provided OperatorHub sources reside. Instead, the Red Hat OpenShift AI Operator can be deployed to a disconnected environment using a private registry to mirror the images.

Installing OpenShift AI in a disconnected environment involves the following high-level tasks:

1. Confirm that your OpenShift Container Platform cluster meets all requirements. See [Requirements for OpenShift AI Self-Managed](#).
2. Add administrative users for OpenShift Container Platform. See [Adding administrative users for OpenShift Container Platform](#).
3. Mirror images to a private registry. See [Mirroring images to a private registry for a disconnected installation](#).
4. Install the Red Hat OpenShift AI Operator. See [Installing the Red Hat OpenShift AI Operator](#).
5. Install OpenShift AI components. See [Installing and managing Red Hat OpenShift AI components](#).
6. Configure user and administrator groups to provide user access to OpenShift AI. See [Adding users](#).
7. Provide your users with the URL for the OpenShift Container Platform cluster on which you deployed OpenShift AI. See [Accessing the OpenShift AI dashboard](#).

### 3.1. REQUIREMENTS FOR OPENSIFT AI SELF-MANAGED

You must meet the following requirements before you can install Red Hat OpenShift AI on your Red Hat OpenShift Container Platform cluster in a disconnected environment:

#### Product subscriptions

- You must have a subscription for Red Hat OpenShift AI Self-Managed.

Contact your Red Hat account manager to purchase new subscriptions. If you do not yet have an account manager, complete the form at <https://www.redhat.com/en/contact> to request one.

#### Cluster administrator access to your OpenShift cluster

- You must have an OpenShift cluster with cluster administrator access. Use an existing cluster or create a cluster by following the OpenShift Container Platform documentation: [Disconnected installation mirroring](#).
- After you install a cluster, configure the Cluster Samples Operator by following the OpenShift Container Platform documentation: [Configuring Samples Operator for a restricted cluster](#).
- Your cluster must have at least 2 worker nodes with at least 8 CPUs and 32 GiB RAM available for OpenShift AI to use when you install the Operator. To ensure that OpenShift AI is usable, additional cluster resources are required beyond the minimum requirements.

- Your cluster is configured with a default storage class that can be dynamically provisioned. Confirm that a default storage class is configured by running the **oc get storageclass** command. If no storage classes are noted with **(default)** beside the name, follow the OpenShift Container Platform documentation to configure a default storage class: [Changing the default storage class](#). For more information about dynamic provisioning, see [Dynamic provisioning](#).
- Open Data Hub must not be installed on the cluster.

For more information about managing the machines that make up an OpenShift cluster, see [Overview of machine management](#).

### An identity provider configured for OpenShift Container Platform

- Red Hat OpenShift AI uses the same authentication systems as Red Hat OpenShift Container Platform. See [Understanding identity provider configuration](#) for more information on configuring identity providers.
- Access to the cluster as a user with the **cluster-admin** role; the **kubeadmin** user is not allowed.

### Internet access on the mirroring machine

Along with Internet access, the following domains must be accessible to mirror images required for the OpenShift AI Self-Managed installation:

- [cdn.redhat.com](https://cdn.redhat.com)
- [subscription.rhn.redhat.com](https://subscription.rhn.redhat.com)
- [registry.access.redhat.com](https://registry.access.redhat.com)
- [registry.redhat.io](https://registry.redhat.io)
- [quay.io](https://quay.io)
  - For CUDA-based images, the following domains must be accessible:
- [ngc.download.nvidia.cn](https://ngc.download.nvidia.cn)
- [developer.download.nvidia.com](https://developer.download.nvidia.com)

### Data science pipelines preparation

- Data Science Pipelines (DSP) 2.0 contains an installation of Argo Workflows. If there is an existing installation of Argo Workflows that is not installed by DSP on your cluster, data science pipelines will be disabled after you install OpenShift AI. Before installing OpenShift AI, ensure that your cluster does not have an existing installation of Argo Workflows that is not installed by DSP, or remove the separate installation of Argo Workflows from your cluster.
- Before you can execute a pipeline in a disconnected environment, you must upload the images to your private registry. For more information, see [Mirroring images to run pipelines in a restricted environment](#).
- You can store your pipeline artifacts in an S3-compatible object storage bucket so that you do not consume local storage. To do this, you must first configure write access to your S3 bucket on your storage account.

### Install KServe dependencies

- To support the KServe component, which is used by the single-model serving platform to serve large models, you must also install Operators for Red Hat OpenShift Serverless and Red Hat OpenShift Service Mesh and perform additional configuration. For more information, see [Serving large models](#).
- If you want to add an authorization provider for the single-model serving platform, you must install the **Red Hat - Authorino** Operator. For information, see [Adding an authorization provider for the single-model serving platform](#).

### Access to object storage

- Components of OpenShift AI require or can use S3-compatible object storage such as AWS S3, MinIO, Ceph, or IBM Cloud Storage. An object store is a data storage mechanism that enables users to access their data either as an object or as a file. The S3 API is the recognized standard for HTTP-based access to object storage services.
- The object storage must be accessible to your OpenShift Container Platform cluster. Deploy the object storage on the same disconnected network as your cluster.
- Object storage is required for the following components:
  - Single- or multi-model serving platforms, to deploy stored models. See [Deploying models on the single-model serving platform](#) or [Deploying a model by using the multi-model serving platform](#).
  - Data science pipelines, to store artifacts, logs, and intermediate results. See [Configuring a pipeline server](#) and [About pipeline logs](#).
- Object storage can be used by the following components:
  - Workbenches, to access large datasets. See [Adding a data connection to your data science project](#).
  - Distributed workloads, to pull input data from and push results to. See [Running distributed data science workloads from data science pipelines](#).
  - Code executed inside a pipeline. For example, to store the resulting model in object storage. See [Overview of pipelines in Jupyterlab](#).

## 3.2. ADDING ADMINISTRATIVE USERS FOR OPENSIFT CONTAINER PLATFORM

Before you can install and configure OpenShift AI for your data scientist users, you must define administrative users. Only users with cluster administrator privileges can install and configure OpenShift AI.

For more information about creating a cluster admin user, see [Creating a cluster admin](#) in the OpenShift Container Platform documentation.

## 3.3. MIRRORING IMAGES TO A PRIVATE REGISTRY FOR A DISCONNECTED INSTALLATION

You can install the Red Hat OpenShift AI Operator to your OpenShift cluster in a disconnected environment by mirroring the required container images to a private container registry. After mirroring the images to a container registry, you can install Red Hat OpenShift AI Operator using OperatorHub.

You can use the *mirror registry for Red Hat OpenShift*, a small-scale container registry that you can use as a target for mirroring the required container images for OpenShift AI in a disconnected environment. Use of the mirror registry for Red Hat OpenShift is optional if another container registry is already available in your installation environment.

## Prerequisites

- You have cluster administrator access to a running OpenShift Container Platform cluster, version 4.12 or greater.
- You have credentials for Red Hat OpenShift Cluster Manager (<https://console.redhat.com/openshift/>).
- Your mirroring machine is running Linux, has 100 GB of space available, and has access to the Internet so that it can obtain the images to populate the mirror repository.
- You have installed the OpenShift CLI (**oc**).
- If you plan to use NVIDIA GPUs, you have mirrored and deployed the NVIDIA GPU Operator. See [Configuring the NVIDIA GPU Operator](#) in the OpenShift Container Platform documentation.
- If you plan to use data science pipelines, you have mirrored the OpenShift Pipelines operator.
- If you plan to use the single-model serving platform to serve large models, you have mirrored the Operators for Red Hat OpenShift Serverless and Red Hat OpenShift Service Mesh. For more information, see [Serving large models](#).
- If you plan to use the distributed workloads component, you have mirrored the Ray cluster image.

## Procedure

1. Create a mirror registry. See [Creating a mirror registry with mirror registry for Red Hat OpenShift](#) in the OpenShift Container Platform documentation.
2. To mirror registry images, install the **oc-mirror** OpenShift CLI plug-in (version 4.12 or greater) on your mirroring machine running Linux. See [Installing the oc-mirror OpenShift CLI plug-in](#) in the OpenShift Container Platform documentation.



### IMPORTANT

Versions of **oc-mirror** before version 4.12 do not allow you to mirror the full image set configuration provided.

3. Create a container image registry credentials file that allows mirroring images from Red Hat to your mirror. See [Configuring credentials that allow images to be mirrored](#) in the OpenShift Container Platform documentation.
4. Open the example image set configuration file (**rhoai-<version>.md**) from the [disconnected installer helper](#) repository and examine its contents.
5. Using the example image set configuration file, create a file called **imageset-config.yaml** and populate it with values suitable for the image set configuration in your deployment.

- To view a list of the available OpenShift versions, run the following command. This might take several minutes. If the command returns errors, repeat the steps in [Configuring credentials that allow images to be mirrored](#).

```
oc-mirror list operators
```

- To see the available channels for a package in a specific version of OpenShift Container Platform (for example, 4.15), run the following command:

```
oc-mirror list operators --catalog=registry.redhat.io/redhat/redhat-operator-index:v4.15 --package=<package-name>
```

- For information about subscription update channels, see [Understanding update channels](#).



### IMPORTANT

The example image set configurations are for demonstration purposes only and might need further alterations depending on your deployment.

To identify the attributes most suitable for your deployment, examine the documentation and use cases in [Mirroring images for a disconnected installation using the oc-mirror plugin](#).

Your **imageset-config.yaml** should look similar to the following example, where **openshift-pipelines-operator-rh** is required for data science pipelines, and both **serverless-operator** and **servicemeshoperator** are required for the KServe component.

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4
storageConfig:
  registry:
    imageURL: registry.example.com:5000/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15
      packages:
        - name: rhods-operator
          channels:
            - name: stable
              minVersion: 2.10.0
              maxVersion: 2.10.0
        - name: openshift-pipelines-operator-rh
          channels:
            - name: stable
        - name: serverless-operator
          channels:
            - name: stable
        - name: servicemeshoperator
          channels:
            - name: stable
```

6. Download the specified image set configuration to a local file on your mirroring machine:



- Replace **mirror-rhoai** with the target directory where you want to output the image set file.
- The target directory path must start with **file://**.
- The download might take several minutes.

```
$ oc mirror --config=./imageset-config.yaml file://mirror-rhoai
```

### TIP

If the **tls: failed to verify certificate: x509: certificate signed by unknown authority** error is returned and you want to ignore it, set **skipTLS** to **true** in your image set configuration file and run the command again.

7. Verify that the image set **.tar** files were created:

```
$ ls mirror-rhoai
mirror_seq1_000000.tar mirror_seq1_000001.tar
```

If an **archiveSize** value was specified in the image set configuration file, the image set might be separated into multiple **.tar** files.

8. Optional: Verify that total size of the image set **.tar** files is around 75 GB:

```
$ du -h --max-depth=1 ./mirror-rhoai/
```

If the total size of the image set is significantly less than 75 GB, run the **oc mirror** command again.

9. Upload the contents of the generated image set to your target mirror registry:

- Replace **mirror-rhoai** with the directory that contains your image set **.tar** files.
- Replace **registry.example.com:5000** with your mirror registry.

```
$ oc mirror --from=./mirror-rhoai docker://registry.example.com:5000
```

### TIP

If the **tls: failed to verify certificate: x509: certificate signed by unknown authority** error is returned and you want to ignore it, run the following command:

```
$ oc mirror --dest-skip-tls --from=./mirror-rhoai docker://registry.example.com:5000
```

10. Log in to your target OpenShift cluster using the OpenShift CLI as a user with the **cluster-admin** role.
11. Verify that the YAML files are present for the **ImageContentSourcePolicy** and **CatalogSource** resources:
  - Replace **<results-directory>** with the name of your results directory.

```
$ ls oc-mirror-workspace/<results-directory>/
```

```
catalogSource-cs-redhat-operator-index.yaml
charts
imageContentSourcePolicy.yaml
mapping.txt
release-signatures
```

12. Install the generated **ImageContentSourcePolicy** and **CatalogSource** resources into the cluster:

- Replace **<results-directory>** with the name of your results directory.

```
$ oc apply -f ./oc-mirror-workspace/<results-directory>/imageContentSourcePolicy.yaml
$ oc apply -f ./oc-mirror-workspace/<results-directory>/catalogSource-cs-redhat-operator-index.yaml
```

### Verification

- Verify that the **CatalogSource** and pod were created successfully:

```
$ oc get catalogsource,pod -n openshift-marketplace
```

This should return at least one catalog and two pods.

- Check that the Red Hat OpenShift AI Operator exists in the OperatorHub:
  - a. Log in to the OpenShift Container Platform cluster web console.
  - b. Click **Operators** → **OperatorHub**.  
The **OperatorHub** page opens.
  - c. Confirm that the Red Hat OpenShift AI Operator is shown.
- If you mirrored additional operators, such as OpenShift Pipelines, Red Hat OpenShift Serverless, or Red Hat OpenShift Service Mesh, check that those operators exist the OperatorHub.

### Additional resources

[Disconnected installation mirroring](#)

## 3.4. INSTALLING THE RED HAT OPENSIFT AI OPERATOR

This section shows how to install the Red Hat OpenShift AI Operator on your OpenShift Container Platform cluster using the command-line interface (CLI) and the OpenShift web console.



### NOTE

If you want to upgrade from a previous version of OpenShift AI rather than performing a new installation, see [Upgrading OpenShift AI in a disconnected environment](#).

**NOTE**

If your OpenShift cluster uses a proxy to access the Internet, you can configure the proxy settings for the Red Hat OpenShift AI Operator. See [Overriding proxy settings of an Operator](#) for more information.

### 3.4.1. Installing the Red Hat OpenShift AI Operator by using the CLI

The following procedure shows how to use the OpenShift command-line interface (CLI) to install the Red Hat OpenShift AI Operator on your OpenShift Container Platform cluster. You must install the Operator before you can install OpenShift AI components on the cluster.

#### Prerequisites

- You have a running OpenShift Container Platform cluster, version 4.12 or greater, configured with a default storage class that can be dynamically provisioned.
- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have downloaded and installed the OpenShift command-line interface (CLI). See [Installing the OpenShift CLI](#).
- You have mirrored the required container images to a private registry. See [Mirroring images to a private registry for a disconnected installation](#).

#### Procedure

1. Open a new terminal window.
2. Follow these steps to log in to your OpenShift Container Platform cluster as a cluster administrator:
  - a. In the upper-right corner of the OpenShift web console, click your user name and select **Copy login command**.
  - b. After you have logged in, click **Display token**.
  - c. Copy the **Log in with this token** command and paste it in the OpenShift command-line interface (CLI).

```
$ oc login --token=<token> --server=<openshift_cluster_url>
```

3. Create a namespace for installation of the Operator by performing the following actions:
  - a. Create a namespace YAML file, for example, **rhods-operator-namespace.yaml**.

```
apiVersion: v1
kind: Namespace
metadata:
  name: redhat-ods-operator 1
```

**1** **redhat-ods-operator** is the recommended namespace for the Operator.

- b. Create the namespace in your OpenShift Container Platform cluster.

■

```
$ oc create -f rhods-operator-namespace.yaml
```

You see output similar to the following:

```
namespace/redhat-ods-operator created
```

4. Create an operator group for installation of the Operator by performing the following actions:
  - a. Create an **OperatorGroup** object custom resource (CR) file, for example, **rhods-operator-group.yaml**.

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: rhods-operator
  namespace: redhat-ods-operator 1
```

- 1** You must specify the same namespace that you created earlier in this procedure.

- b. Create the **OperatorGroup** object in your OpenShift Container Platform cluster.

```
$ oc create -f rhods-operator-group.yaml
```

You see output similar to the following:

```
operatorgroup.operators.coreos.com/rhods-operator created
```

5. Create a subscription for installation of the Operator by performing the following actions:
  - a. Create a **Subscription** object CR file, for example, **rhods-operator-subscription.yaml**.

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: rhods-operator
  namespace: redhat-ods-operator 1
spec:
  name: rhods-operator
  channel: stable 2
  source: cs-redhat-operator-index
  sourceNamespace: openshift-marketplace
```

- 1** You must specify the same namespace that you created earlier in this procedure.

- 2** For **channel**, select a value of **fast**, **stable**, **stable-x.y**, **eus-x.y**, or **alpha**. For more information about selecting an update channel, see [Understanding update channels](#).

- b. Create the **Subscription** object in your OpenShift Container Platform cluster to install the Operator.

```
$ oc create -f rhods-operator-subscription.yaml
```

You see output similar to the following:

```
subscription.operators.coreos.com/rhods-operator created
```

### Verification

- In the OpenShift Container Platform web console, click **Operators** → **Installed Operators** and confirm that the Red Hat OpenShift AI Operator shows one of the following statuses:
  - **Installing** - installation is in progress; wait for this to change to **Succeeded**. This might take several minutes.
  - **Succeeded** - installation is successful.
- In the web console, click **Home** → **Projects** and confirm that the following project namespaces are visible and listed as **Active**:
  - **redhat-ods-applications**
  - **redhat-ods-monitoring**
  - **redhat-ods-operator**

### Additional resources

- [Installing and managing Red Hat OpenShift AI components](#)
- [Adding users](#)
- [Adding Operators to a cluster](#)

## 3.4.2. Installing the Red Hat OpenShift AI Operator by using the web console

The following procedure shows how to use the OpenShift Container Platform web console to install the Red Hat OpenShift AI Operator on your cluster. You must install the Operator before you can install OpenShift AI components on the cluster.

### Prerequisites

- You have a running OpenShift Container Platform cluster, version 4.12 or greater, configured with a default storage class that can be dynamically provisioned.
- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have mirrored the required container images to a private registry. See [Mirroring images to a private registry for a disconnected installation](#).

### Procedure

1. Log in to the OpenShift Container Platform web console as a cluster administrator.
2. In the web console, click **Operators** → **OperatorHub**.
3. On the **OperatorHub** page, locate the Red Hat OpenShift AI Operator by scrolling through the available Operators or by typing *Red Hat OpenShift AI* into the **Filter by keyword** box.

4. Click the **Red Hat OpenShift AI** tile. The **Red Hat OpenShift AI** information pane opens.
5. Select a **Channel**. For information about subscription update channels, see [Understanding update channels](#).
6. Select a **Version**.
7. Click **Install**. The **Install Operator** page opens.
8. Review or change the selected channel and version as needed.
9. For **Installation mode**, note that the only available value is **All namespaces on the cluster (default)**. This installation mode makes the Operator available to all namespaces in the cluster.
10. For **Installed Namespace**, select **Operator recommended Namespace: redhat-ods-operator**.
11. For **Update approval**, select one of the following update strategies:
  - **Automatic**: Your environment attempts to install new updates when they are available based on the content of your mirror.
  - **Manual**: A cluster administrator must approve any new updates before installation begins.



### IMPORTANT

By default, the Red Hat OpenShift AI Operator follows a sequential update process. This means that if there are several versions between the current version and the target version, Operator Lifecycle Manager (OLM) upgrades the Operator to each of the intermediate versions before it upgrades it to the final, target version.

If you configure automatic upgrades, OLM automatically upgrades the Operator to the *latest* available version. If you configure manual upgrades, a cluster administrator must manually approve each sequential update between the current version and the final, target version.

For information about supported versions, see [Red Hat OpenShift AI Life Cycle](#).

12. Click **Install**.  
The **Installing Operators** pane appears. When the installation finishes, a checkmark appears next to the Operator name.

### Verification

- In the OpenShift Container Platform web console, click **Operators** → **Installed Operators** and confirm that the Red Hat OpenShift AI Operator shows one of the following statuses:
  - **Installing** - installation is in progress; wait for this to change to **Succeeded**. This might take several minutes.
  - **Succeeded** - installation is successful.
- In the web console, click **Home** → **Projects** and confirm that the following project namespaces are visible and listed as **Active**:
  - **redhat-ods-applications**

- **redhat-ods-monitoring**
- **redhat-ods-operator**

#### Additional resources

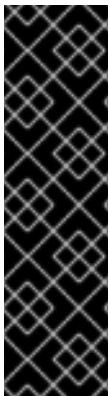
- [Installing and managing Red Hat OpenShift AI components](#)
- [Adding users](#)
- [Adding Operators to a cluster](#)

## 3.5. INSTALLING AND MANAGING RED HAT OPENSIFT AI COMPONENTS

The following procedures show how to use the command-line interface (CLI) and OpenShift Container Platform web console to install and manage components of Red Hat OpenShift AI on your OpenShift Container Platform cluster.

### 3.5.1. Installing Red Hat OpenShift AI components by using the CLI

The following procedure shows how to use the OpenShift command-line interface (CLI) to install specific components of Red Hat OpenShift AI on your OpenShift Container Platform cluster.



#### IMPORTANT

The following procedure describes how to create and configure a **DataScienceCluster** object to install Red Hat OpenShift AI components as part of a *new* installation. However, if you upgraded from version 1 of OpenShift AI (previously OpenShift Data Science), the upgrade process automatically created a default **DataScienceCluster** object. If you upgraded from a previous minor version, the upgrade process uses the settings from the previous version's **DataScienceCluster** object. To inspect the **DataScienceCluster** object and change the installation status of Red Hat OpenShift AI components, see [Updating the installation status of Red Hat OpenShift AI components by using the web console](#).

#### Prerequisites

- The Red Hat OpenShift AI Operator is installed on your OpenShift Container Platform cluster. See [Installing the Red Hat OpenShift AI Operator](#).
- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have downloaded and installed the OpenShift command-line interface (CLI). See [Installing the OpenShift CLI](#).

#### Procedure

1. Open a new terminal window.
2. Follow these steps to log in to your OpenShift Container Platform cluster as a cluster administrator:

- a. In the upper-right corner of the OpenShift web console, click your user name and select **Copy login command**.
- b. After you have logged in, click **Display token**.
- c. Copy the **Log in with this token** command and paste it in the OpenShift command-line interface (CLI).

```
$ oc login --token=<token> --server=<openshift_cluster_url>
```

3. Create a **DataScienceCluster** object custom resource (CR) file, for example, **rhods-operator-dsc.yaml**.

```
apiVersion: datasciencecluster.opendatahub.io/v1
kind: DataScienceCluster
metadata:
  name: default-dsc
spec:
  components:
    codeflare:
      managementState: Removed
    dashboard:
      managementState: Removed
    datasciencepipelines:
      managementState: Removed
    kserve:
      managementState: Removed 1 2
    kueue:
      managementState: Removed
    modelmeshserving:
      managementState: Removed
    ray:
      managementState: Removed
    workbenches:
      managementState: Removed
```

- 1 To fully install the KServe component, which is used by the single-model serving platform to serve large models, you must install Operators for Red Hat OpenShift Service Mesh and Red Hat OpenShift Serverless and perform additional configuration. See [Serving large models](#).
- 2 If you have *not* enabled the KServe component (that is, you set the value of the **managementState** field to **Removed**), you must also disable the dependent Service Mesh component to avoid errors. See [Disabling KServe dependencies](#).

4. In the **spec.components** section of the CR, for each OpenShift AI component shown, set the value of the **managementState** field to either **Managed** or **Removed**. These values are defined as follows:

#### Managed

The Operator actively manages the component, installs it, and tries to keep it active. The Operator will upgrade the component only if it is safe to do so.

#### Removed



The Operator actively manages the component but does not install it. If the component is already installed, the Operator will try to remove it.



### IMPORTANT

- To learn how to fully install the KServe component, which is used by the single-model serving platform to serve large models, see [Serving large models](#).
- To learn how to configure the distributed workloads feature that uses the CodeFlare, KubeRay, and Kueue components, see [Configuring distributed workloads](#).
- To learn how to run distributed workloads in a disconnected environment, see [Running distributed data science workloads in a disconnected environment](#).

5. Create the **DataScienceCluster** object in your OpenShift Container Platform cluster to install the specified OpenShift AI components.

```
$ oc create -f rhods-operator-dsc.yaml
```

You see output similar to the following:

```
datasciencecluster.datasciencecluster.opendatahub.io/default created
```

### Verification

- Confirm that there is a running pod for each component:
  1. In the OpenShift Container Platform web console, click **Workloads** → **Pods**.
  2. In the **Project** list at the top of the page, select **redhat-ods-applications**.
  3. In the applications namespace, confirm that there are running pods for each of the OpenShift AI components that you installed.
- Confirm the status of all installed components:
  1. In the OpenShift Container Platform web console, click **Operators** → **Installed Operators**.
  2. Click the Red Hat OpenShift AI Operator.
  3. Click the **Data Science Cluster** tab and select the **DataScienceCluster** object called **default-dsc**.
  4. Select the **YAML** tab.
  5. In the **installedComponents** section, confirm that the components you installed have a status value of **true**.



### NOTE

If a component shows with the **component-name: {}** format in the **spec.components** section of the CR, the component is not installed.

### 3.5.2. Installing Red Hat OpenShift AI components by using the web console

The following procedure shows how to use the OpenShift Container Platform web console to install specific components of Red Hat OpenShift AI on your cluster.



#### IMPORTANT

The following procedure describes how to create and configure a **DataScienceCluster** object to install Red Hat OpenShift AI components as part of a *new* installation. However, if you upgraded from version 1 of OpenShift AI (previously OpenShift Data Science), the upgrade process automatically created a default **DataScienceCluster** object. If you upgraded from a previous minor version, the upgrade process used the settings from the previous version's **DataScienceCluster** object. To inspect the **DataScienceCluster** object and change the installation status of Red Hat OpenShift AI components, see [Updating the installation status of Red Hat OpenShift AI components by using the web console](#).

#### Prerequisites

- The Red Hat OpenShift AI Operator is installed on your OpenShift Container Platform cluster. See [Installing the Red Hat OpenShift AI Operator](#).
- You have cluster administrator privileges for your OpenShift Container Platform cluster.

#### Procedure

1. Log in to the OpenShift Container Platform web console as a cluster administrator.
2. In the web console, click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
3. Create a **DataScienceCluster** object to install OpenShift AI components by performing the following actions:
  - a. Click the **Data Science Cluster** tab.
  - b. Click **Create DataScienceCluster**.
  - c. For **Configure via**, select **YAML view**.  
An embedded YAML editor opens showing a default custom resource (CR) for the **DataScienceCluster** object.
  - d. In the **spec.components** section of the CR, for each OpenShift AI component shown, set the value of the **managementState** field to either **Managed** or **Removed**. These values are defined as follows:

##### Managed

The Operator actively manages the component, installs it, and tries to keep it active. The Operator will upgrade the component only if it is safe to do so.

##### Removed

The Operator actively manages the component but does not install it. If the component is already installed, the Operator will try to remove it.



### IMPORTANT

- To learn how to install the KServe component, which is used by the single-model serving platform to serve large models, see [Serving large models](#).
- If you have *not* enabled the KServe component (that is, you set the value of the **managementState** field to **Removed**), you must also disable the dependent Service Mesh component to avoid errors. See [Disabling KServe dependencies](#).
- To learn how to configure the distributed workloads feature that uses the CodeFlare and KubeRay components, see [Configuring distributed workloads](#).

4. Click **Create**.

### Verification

- Confirm that there is a running pod for each component:
  1. In the OpenShift Container Platform web console, click **Workloads** → **Pods**.
  2. In the **Project** list at the top of the page, select **redhat-ods-applications**.
  3. In the applications namespace, confirm that there are running pods for each of the OpenShift AI components that you installed.
- Confirm the status of all installed components:
  1. In the OpenShift Container Platform web console, click **Operators** → **Installed Operators**.
  2. Click the Red Hat OpenShift AI Operator.
  3. Click the **Data Science Cluster** tab and select the **DataScienceCluster** object called **default-dsc**.
  4. Select the **YAML** tab.
  5. In the **installedComponents** section, confirm that the components you installed have a status value of **true**.



### NOTE

If a component shows with the **component-name: {}** format in the **spec.components** section of the CR, the component is not installed.

### 3.5.3. Updating the installation status of Red Hat OpenShift AI components by using the web console

The following procedure shows how to use the OpenShift Container Platform web console to update the installation status of components of Red Hat OpenShift AI on your OpenShift Container Platform cluster.



## IMPORTANT

If you upgraded from version 1 to version 2 of OpenShift AI, the upgrade process automatically created a default **DataScienceCluster** object and enabled several components of OpenShift AI. If you upgraded from a previous minor version, the upgrade process used the settings from the previous version's **DataScienceCluster** object.

The following procedure describes how to edit the **DataScienceCluster** object to do the following:

- Change the installation status of the existing Red Hat OpenShift AI components
- Add additional components to the **DataScienceCluster** object that were not available in the previous version of OpenShift AI.

## Prerequisites

- The Red Hat OpenShift AI Operator is [installed](#) on your OpenShift Container Platform cluster.
- You have cluster administrator privileges for your OpenShift Container Platform cluster.

## Procedure

1. Log in to the OpenShift Container Platform web console as a cluster administrator.
2. In the web console, click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
3. Click the **Data Science Cluster** tab.
4. On the **DataScienceClusters** page, click the **default** object.
5. Click the **YAML** tab.  
An embedded YAML editor opens showing the custom resource (CR) file for the **DataScienceCluster** object.
6. In the **spec.components** section of the CR, for each OpenShift AI component shown, set the value of the **managementState** field to either **Managed** or **Removed**. These values are defined as follows:



## NOTE

If a component shows with the **component-name: {}** format in the **spec.components** section of the CR, the component is not installed.

### Managed

The Operator actively manages the component, installs it, and tries to keep it active. The Operator will upgrade the component only if it is safe to do so.

### Removed

The Operator actively manages the component but does not install it. If the component is already installed, the Operator will try to remove it.



## IMPORTANT

- To learn how to install the KServe component, which is used by the single-model serving platform to serve large models, see [Serving large models](#).
- If you have *not* enabled the KServe component (that is, you set the value of the **managementState** field to **Removed**), you must also disable the dependent Service Mesh component to avoid errors. See [Disabling KServe dependencies](#).
- If they are not already present in the CR file, you can install the CodeFlare, KubeRay, and Kueue components by adding the **codeflare**, **ray**, and **kueue** entries to the **spec.components** section of the CR and setting the **managementState** field for the components to **Managed**.
- To learn how to configure the distributed workloads feature that uses the CodeFlare, KubeRay, and Kueue components, see [Configuring distributed workloads](#).
- To learn how to run distributed workloads in a disconnected environment, see [Running distributed data science workloads in a disconnected environment](#).

### 7. Click **Save**.

For any components that you updated, OpenShift AI initiates a rollout that affects all pods to use the updated image.

## Verification

- Confirm that there is a running pod for each component:
  1. In the OpenShift Container Platform web console, click **Workloads** → **Pods**.
  2. In the **Project** list at the top of the page, select **redhat-ods-applications**.
  3. In the applications namespace, confirm that there are running pods for each of the OpenShift AI components that you installed.
- Confirm the status of all installed components:
  1. In the OpenShift Container Platform web console, click **Operators** → **Installed Operators**.
  2. Click the Red Hat OpenShift AI Operator.
  3. Click the **Data Science Cluster** tab and select the **DataScienceCluster** object called **default-dsc**.
  4. Select the **YAML** tab.
  5. In the **installedComponents** section, confirm that the components you installed have a status value of **true**.



## NOTE

If a component shows with the **component-name: {}** format in the **spec.components** section of the CR, the component is not installed.

### 3.5.4. Disabling KServe dependencies

If you have *not* enabled the KServe component (that is, you set the value of the **managementState** field to **Removed**), you must also disable the dependent Service Mesh component to avoid errors.

#### Prerequisites

- You have used the OpenShift command-line interface (CLI) or web console to disable the KServe component.

#### Procedure

1. Log in to the OpenShift web console as a cluster administrator.
2. In the web console, click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
3. Disable the OpenShift Service Mesh component as follows:
  - a. Click the **DSC Initialization** tab.
  - b. Click the **default-dsci** object.
  - c. Click the **YAML** tab.
  - d. In the **spec** section, add the **serviceMesh** component (if it is not already present) and configure the **managementState** field as shown:

```
spec:
  serviceMesh:
    managementState: Removed
```

- e. Click **Save**.

#### Verification

1. In the web console, click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.  
The Operator details page opens.
2. In the **Conditions** section, confirm that there is no **ReconcileComplete** condition with a status value of **Unknown**.

## 3.6. CONFIGURING THE OPENSIFT AI OPERATOR LOGGER

You can adjust the log level of OpenShift AI Operator components to increase or reduce log verbosity to suit your use case.

### 3.6.1. Configuring the OpenShift AI Operator logger

You can change the log level for OpenShift AI Operator components by setting the **.spec.devFlags.logmode** flag for the **DSC Initialization/DSCI** custom resource during runtime. If you do not set a **logmode** value, the logger uses the INFO log level by default.

The log level that you set with `.spec.devFlags.logmode` applies to all components, not just those in a **Managed** state.

The following table shows the available log levels:

Log level	Stacktrace level	Verbosity	Output	Timestamp type
<b>devel</b> or <b>development</b>	WARN	INFO	Console	Epoch timestamps
"" (or no <b>logmode</b> value set)	ERROR	INFO	JSON	Human-readable timestamps
<b>prod</b> or <b>production</b>	ERROR	INFO	JSON	Human-readable timestamps

Logs that are set to **devel** or **development** generate in a plain text console format. Logs that are set to **prod**, **production**, or which do not have a level set generate in a JSON format.

### Prerequisites

- You have admin access to the **DSCInitialization** resources in the OpenShift Container Platform cluster.
- You installed the OpenShift command line interface (**oc**) as described in [Get Started with the CLI](#).

### Procedure

1. Log in to the OpenShift Container Platform as a cluster administrator.
2. Click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
3. Click the **DSC Initialization** tab.
4. Click the **default-dsci** object.
5. Click the **YAML** tab.
6. In the **spec** section, update the `.spec.devFlags.logmode` flag with the log level that you want to set.

```
apiVersion: dscinitialization.opendatahub.io/v1
kind: DSCInitialization
metadata:
  name: default-dsci
spec:
  devFlags:
    logmode: development
```

7. Click **Save**.

You can also configure the log level from the OpenShift CLI by using the following command with the **logmode** value set to the log level that you want.

```
oc patch dsci default-dsci -p '{"spec":{"devFlags":{"logmode":"development"}}}' --type=merge
```

### Verification

- If you set the component log level to **devel** or **development**, logs generate more frequently and include logs at **WARN** level and above.
- If you set the component log level to **prod** or **production**, or do not set a log level, logs generate less frequently and include logs at **ERROR** level or above.

#### 3.6.1.1. Viewing the OpenShift AI Operator log

1. Log in to the OpenShift CLI.
2. Run the following command:

```
oc get pods -l name=rhods-operator -o name -n redhat-ods-operator | xargs -l {} oc logs -f {}  
-n redhat-ods-operator
```

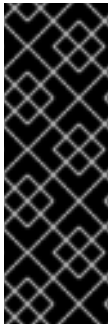
The operator pod log opens.

You can also view the operator pod log in the OpenShift Console, under **Workloads > Deployments > Pods > redhat-ods-operator > Logs**.



## CHAPTER 4. PREPARING OPENSIFT AI FOR USE IN IBM CLOUD PAK FOR DATA

The procedures in this section show how to prepare Red Hat OpenShift AI for use in IBM Cloud Pak for Data version 5.0 or greater. These versions of Cloud Pak for Data include `watsonx.ai`.



### IMPORTANT

The procedures in this section apply when you want to prepare a *new* installation of Red Hat OpenShift AI for use in IBM Cloud Pak for Data version 5.0 or greater. The procedures show how to install the KServe component of OpenShift AI in *raw* mode, which means that KServe does not have any other components as dependencies. However, if you have an existing deployment of OpenShift AI with KServe and its dependencies already installed and enabled, you *do not* need to modify the configuration for use in Cloud Pak for Data.

### 4.1. INSTALLING THE RED HAT OPENSIFT AI OPERATOR BY USING THE CLI



### IMPORTANT

Follow this procedure *only* if you are preparing Red Hat OpenShift AI for use in IBM Cloud Pak for Data version 5.0 or greater. These versions of Cloud Pak for Data include `watsonx.ai`. If this use case does not apply to your organization, see [Installing and deploying OpenShift AI in a disconnected environment](#) for more generally applicable instructions.

This procedure shows how to use the OpenShift command-line interface (CLI) to install the Red Hat OpenShift AI Operator on your OpenShift Container Platform cluster. You must install the Operator before you can manage the installation of OpenShift AI components.

#### Prerequisites

- You have a running OpenShift Container Platform cluster, version 4.12 or greater, configured with a default storage class that can be dynamically provisioned.
- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have downloaded and installed the OpenShift command-line interface (CLI). See [Installing the OpenShift CLI](#).
- You have mirrored the required container images to a private registry. See [Mirroring images to a private registry for a disconnected installation](#).

#### Procedure

1. Open a new terminal window.
2. In the OpenShift command-line interface (CLI), log in to your OpenShift Container Platform cluster as a cluster administrator, as shown in the following example:

```
$ oc login <openshift_cluster_url> -u <admin_username> -p <password>
```

3. Create a new YAML file with the following contents:

```

apiVersion: v1
kind: Namespace 1
metadata:
  name: redhat-ods-operator
---
apiVersion: operators.coreos.com/v1
kind: OperatorGroup 2
metadata:
  name: rhods-operator
  namespace: redhat-ods-operator
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription 3
metadata:
  name: rhods-operator
  namespace: redhat-ods-operator
spec:
  name: rhods-operator
  channel: eus-2.8 4
  source: cs-redhat-operator-index
  sourceNamespace: openshift-marketplace
  config:
    env:
      - name: "DISABLE_DSC_CONFIG"

```

- 1** Defines the recommended **redhat-ods-operator** namespace for installation of the Operator.
- 2** Defines an Operator group for installation of the Operator. You must specify the same namespace that you defined earlier in the file. Also, another Operator group must not exist in the namespace.
- 3** Defines a subscription for the Operator. You must specify the same namespace that you defined earlier in the file.
- 4** For **channel**, specify a value of **eus-2.8**. This channel name indicates that OpenShift AI 2.8 is an Extended Update Support (EUS) version. For more information, see [Red Hat OpenShift AI Self-Managed Life Cycle](#).

4. Deploy the YAML file to create the namespace, Operator group, and subscription that you defined.

```
$ oc create -f <file_name>.yaml
```

You see output similar to the following:

```

namespace/redhat-ods-operator created
operatorgroup.operators.coreos.com/rhods-operator created
subscription.operators.coreos.com/rhods-operator created

```

5. Create another new YAML file with the following contents:

```

apiVersion: dscinitialization.opendatahub.io/v1
kind: DSCInitialization 1
metadata:
  name: default-dsci
spec:
  applicationsNamespace: redhat-ods-applications
  monitoring:
    managementState: Managed
    namespace: redhat-ods-monitoring
  serviceMesh:
    managementState: Removed 2
  trustedCABundle:
    managementState: Managed
    customCABundle: ""

```

- 1** Defines a **DSCInitialization** object called **default-dsci**. The **DSCInitialization** object is used by the Operator to manage resources that are common to all product components.
- 2** Specifies that the **serviceMesh** component (which is a dependency for KServe in some setups) is *not* installed. This setting is required when preparing OpenShift AI for use in IBM products.

## Verification

- In the OpenShift CLI, check that there is a running pod for the Operator by performing the following actions:
  - Check the pods in the **redhat-ods-operator** project.

```
$ oc get pods -n redhat-ods-operator
```

- Confirm that there is a **rhods-operator-\*** pod with a **Running** status, as shown in the following example:

```

NAME                                READY STATUS  RESTARTS  AGE
rhods-operator-56c85d44c9-vtk74    1/1   Running  0         3h57m

```

- In the OpenShift CLI, check that the **DSCInitialization** object that you created is running by performing the following actions:
  - Check the cluster for **DSCInitialization** objects.

```
$ oc get dscinitialization
```

- Confirm that there is a **default-dsci** object with a **Ready** status, as shown in the following example:

```

NAME      AGE   PHASE
default-dsci 4d18h Ready

```

## Additional resources

- [Adding Operators to a cluster](#)

## 4.2. MANAGING RED HAT OPENSIFT AI COMPONENTS BY USING THE CLI

The following procedure shows how to use the OpenShift command-line interface (CLI) to manage the installation of specific components of Red Hat OpenShift AI on your OpenShift Container Platform cluster.



### IMPORTANT

Follow this procedure *only* if you are preparing Red Hat OpenShift AI for use in IBM Cloud Pak for Data version 5.0 or greater. These versions of Cloud Pak for Data include watsonx.ai. If this use case does not apply to your organization, see [Installing and managing Red Hat OpenShift AI components](#) for more generally applicable instructions.

### Prerequisites

- The Red Hat OpenShift AI Operator is installed on your OpenShift Container Platform cluster. See [Installing the Red Hat OpenShift AI Operator by using the CLI](#).
- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have downloaded and installed the OpenShift command-line interface (CLI). See [Installing the OpenShift CLI](#).

### Procedure

1. Open a new terminal window.
2. In the OpenShift command-line interface (CLI), log in to your OpenShift Container Platform cluster as a cluster administrator, as shown in the following example:

```
$ oc login <openshift_cluster_url> -u <admin_username> -p <password>
```

3. Create a new YAML file with the following contents:

```
kind: DataScienceCluster 1
apiVersion: datasciencecluster.opendatahub.io/v1
metadata:
  name: default-dsc
  labels:
    app.kubernetes.io/name: datasciencecluster
    app.kubernetes.io/instance: default-dsc
    app.kubernetes.io/part-of: rhods-operator
    app.kubernetes.io/managed-by: kustomize
    app.kubernetes.io/created-by: rhods-operator
spec:
  components:
    codeflare:
      managementState: Removed
  dashboard:
    managementState: Removed
  datasciencepipelines:
    managementState: Removed
  kserve:
```

```

managementState: Managed
defaultDeploymentMode: RawDeployment 2
serving:
  managementState: Removed 3
  name: knative-serving
kueue:
  managementState: Removed
modelmeshserving:
  managementState: Removed
ray:
  managementState: Removed
workbenches:
  managementState: Removed

```

- 1** Defines a new **DataScienceCluster** object called **default-dsc**. The **DataScienceCluster** object is used by the Operator to manage OpenShift AI components.
- 2** Specifies that the KServe component is installed and managed by the Operator in *raw* mode, which means that KServe does not have any other components as dependencies.
- 3** Specifies that the **serving** component (which is a dependency for KServe in some setups) is *not* installed. This setting is required when preparing OpenShift AI for use in Cloud Pak for Data.

In addition, observe that the value of the **managementState** field for all other OpenShift AI components is set to **Removed**. This means that the components are not installed.

In general, the **Managed** and **Removed** values are described as follows:

#### Managed

The Operator actively manages the component, installs it, and tries to keep it active. The Operator will upgrade the component only if it is safe to do so.

#### Removed

The Operator actively manages the component but does not install it. If the component is already installed, the Operator will try to remove it.

4. Create the **DataScienceCluster** object on your OpenShift cluster.

```
$ oc create -f <file_name>.yaml
```

You see output similar to the following:

```
datasciencecluster.datasciencecluster.opendatahub.io/default-dsc created
```

#### Verification

- In the OpenShift CLI, check that there is a running pod for the KServe controller by performing the following actions:
  - Check the pods in the **redhat-ods-applications** project.

```
$ oc get pods -n redhat-ods-applications
```

- o Confirm that there is a **kserve-controller-manager-\*** pod with a **Running** status, as shown in the following example:

```

NAME                                READY STATUS   RESTARTS AGE
kserve-controller-manager-57796d5b44-sh9n5  1/1   Running    0      4m57s

```

### 4.3. EDITING THE MODEL INFERENCE CONFIGURATION

Particular use cases in IBM Cloud Pak for Data version 5.0 or greater (which include watsonx.ai) might require customizations to the model inferencing configuration used by Red Hat OpenShift AI. Before you can make any such customizations, you need to put the inferencing configuration file in an editable state. In addition, you must make a specific configuration update that prevents errors when using OpenShift AI in Cloud Pak for Data and watsonx.ai.



#### IMPORTANT

Follow this procedure *only* if you are preparing Red Hat OpenShift AI for use in IBM Cloud Pak for Data version 5.0 or greater. These versions of Cloud Pak for Data include watsonx.ai.

#### Prerequisites

- The Red Hat OpenShift AI Operator is installed on your OpenShift Container Platform cluster. See [Installing the Red Hat OpenShift AI Operator by using the CLI](#).
- You have cluster administrator privileges for your OpenShift Container Platform cluster.

#### Procedure

1. Log in to the OpenShift Container Platform web console as a cluster administrator.
2. In the web console, click **Workloads** → **ConfigMaps**.
3. In the **Project** list, click **redhat-ods-applications**.
4. On the list of **ConfigMap** resources, click the **inferenceservice-config** resource and then click the **YAML** tab.
5. In the **metadata.annotations** section of the file, add **opendatahub.io/managed: 'false'** as shown:

```

metadata:
  annotations:
    internal.config.kubernetes.io/previousKinds: ConfigMap
    internal.config.kubernetes.io/previousNames: inferenceservice-config
    internal.config.kubernetes.io/previousNamespaces: opendatahub
    opendatahub.io/managed: 'false'

```

The additional annotation makes the inferencing configuration file editable.

6. To prevent errors when using OpenShift AI in Cloud Pak for Data (including watsonx.ai), update the configuration as follows:
  - a. In the YAML file, locate the following entry:

```
"domainTemplate": "{{ .Name }}-{{ .Namespace }}.{{ .IngressDomain }}"
```

- b. Update the value of the **domainTemplate** field as shown:

```
"domainTemplate": "example.com"
```

This new, explicitly specified value ensures that OpenShift AI cannot generate a value for the **domainTemplate** field that exceeds the maximum length and causes an error. The **domainTemplate** field is not used by the raw deployment mode that you configured for the KServe component when preparing OpenShift AI for use in Cloud Pak for Data.

7. Click **Save**.

## CHAPTER 5. WORKING WITH CERTIFICATES

Certificates are used by various components in OpenShift Container Platform to validate access to the cluster. For clusters that rely on self-signed certificates, you can add those self-signed certificates to a cluster-wide Certificate Authority (CA) bundle and use the CA bundle in Red Hat OpenShift AI. You can also use self-signed certificates in a custom CA bundle that is separate from the cluster-wide bundle. Administrators can add a CA bundle, remove a CA bundle from all namespaces, remove a CA bundle from individual namespaces, or manually manage certificate changes instead of the system.

### 5.1. UNDERSTANDING CERTIFICATES IN OPENSIFT AI

For OpenShift Container Platform clusters that rely on self-signed certificates, you can add those self-signed certificates to a cluster-wide Certificate Authority (CA) bundle (**ca-bundle.crt**) and use the CA bundle in Red Hat OpenShift AI. You can also use self-signed certificates in a custom CA bundle (**odh-ca-bundle.crt**) that is separate from the cluster-wide bundle.

#### 5.1.1. How CA bundles are injected

After installing OpenShift AI, the Red Hat OpenShift AI Operator automatically creates an empty **odh-trusted-ca-bundle** configuration file (ConfigMap), and the Cluster Network Operator (CNO) injects the cluster-wide CA bundle into the **odh-trusted-ca-bundle** configMap with the label "config.openshift.io/inject-trusted-cabundle". The components deployed in the affected namespaces are responsible for mounting this configMap as a volume in the deployment pods.

```
apiVersion: v1
kind: ConfigMap
metadata:
  labels:
    app.kubernetes.io/part-of: opendatahub-operator
    config.openshift.io/inject-trusted-cabundle: 'true'
name: odh-trusted-ca-bundle
```

After the CNO operator injects the bundle, it updates the ConfigMap with the **ca-bundle.crt** file containing the certificates.

```
apiVersion: v1
kind: ConfigMap
metadata:
  labels:
    app.kubernetes.io/part-of: opendatahub-operator
    config.openshift.io/inject-trusted-cabundle: 'true'
name: odh-trusted-ca-bundle
data:
  ca-bundle.crt: |
    <BUNDLE OF CLUSTER-WIDE CERTIFICATES>
```

#### 5.1.2. How the ConfigMap is managed

By default, the Red Hat OpenShift AI Operator manages the **odh-trusted-ca-bundle** ConfigMap. If you want to manage or remove the **odh-trusted-ca-bundle** ConfigMap, or add a custom CA bundle (**odh-ca-bundle.crt**) separate from the cluster-wide CA bundle (**ca-bundle.crt**), you can use the **trustedCABundle** property in the Operator's DSC Initialization (DSCI) object.



```
spec:
  trustedCABundle:
    managementState: Managed
    customCABundle: ""
```

In the Operator's DSCI object, you can set the **spec.trustedCABundle.managementState** field to the following values:

- **Managed:** The Red Hat OpenShift AI Operator manages the **odh-trusted-ca-bundle** ConfigMap and adds it to all non-reserved existing and new namespaces (the ConfigMap is not added to any reserved or system namespaces, such as **default**, **openshift-\*** or **kube-\***). The ConfigMap is automatically updated to reflect any changes made to the **customCABundle** field. This is the default value after installing Red Hat OpenShift AI.
- **Removed:** The Red Hat OpenShift AI Operator removes the **odh-trusted-ca-bundle** ConfigMap (if present) and disables the creation of the ConfigMap in new namespaces. If you change this field from **Managed** to **Removed**, the **odh-trusted-ca-bundle** ConfigMap is also deleted from namespaces. This is the default value after upgrading Red Hat OpenShift AI from 2.7 or earlier versions to 2.10.
- **Unmanaged:** The Red Hat OpenShift AI Operator does not manage the **odh-trusted-ca-bundle** ConfigMap, allowing for an administrator to manage it instead. Changing the **managementState** from **Managed** to **Unmanaged** does not remove the **odh-trusted-ca-bundle** ConfigMap, but the ConfigMap is not updated if you make changes to the **customCABundle** field.

In the Operator's DSCI object, you can add a custom certificate to the **spec.trustedCABundle.customCABundle** field. This adds the **odh-ca-bundle.crt** file containing the certificates to the **odh-trusted-ca-bundle** ConfigMap, as shown in the following example:

```
apiVersion: v1
kind: ConfigMap
metadata:
  labels:
    app.kubernetes.io/part-of:.opendatahub-operator
    config.openshift.io/inject-trusted-cabundle: 'true'
  name: odh-trusted-ca-bundle
data:
  ca-bundle.crt: |
    <BUNDLE OF CLUSTER-WIDE CERTIFICATES>
  odh-ca-bundle.crt: |
    <BUNDLE OF CUSTOM CERTIFICATES>
```

## 5.2. ADDING A CA BUNDLE

There are two ways to add a Certificate Authority (CA) bundle to OpenShift AI. You can use one or both of these methods:

- For OpenShift Container Platform clusters that rely on self-signed certificates, you can add those self-signed certificates to a cluster-wide Certificate Authority (CA) bundle (**ca-bundle.crt**) and use the CA bundle in Red Hat OpenShift AI. To use this method, log in to the OpenShift Container Platform as a cluster administrator and follow the steps as described in [Configuring the cluster-wide proxy during installation](#).

- You can use self-signed certificates in a custom CA bundle (**odh-ca-bundle.crt**) that is separate from the cluster-wide bundle. To use this method, follow the steps in this section.

## Prerequisites

- You have admin access to the **DSCInitialization** resources in the OpenShift Container Platform cluster.
- You installed the OpenShift command line interface (**oc**) as described in [Get Started with the CLI](#).
- You are working in a new installation of Red Hat OpenShift AI. If you upgraded Red Hat OpenShift AI, see [Adding a CA bundle after upgrading](#).

## Procedure

1. Log in to the OpenShift Container Platform.
2. Click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
3. Click the **DSC Initialization** tab.
4. Click the **default-dsci** object.
5. Click the **YAML** tab.
6. In the **spec** section, add the custom certificate to the **customCABundle** field for **trustedCABundle**, as shown in the following example:

```
spec:
  trustedCABundle:
    managementState: Managed
    customCABundle: |
      -----BEGIN CERTIFICATE-----
      examplebundle123
      -----END CERTIFICATE-----
```

7. Click **Save**.

## Verification

- If you are using a cluster-wide CA bundle, run the following command to verify that all non-reserved namespaces contain the **odh-trusted-ca-bundle** ConfigMap:

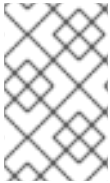
```
$ oc get configmaps --all-namespaces -l app.kubernetes.io/part-of=opendatahub-operator |
grep odh-trusted-ca-bundle
```

- If you are using a custom CA bundle, run the following command to verify that a non-reserved namespace contains the **odh-trusted-ca-bundle** ConfigMap and that the ConfigMap contains your **customCABundle** value. In the following command, *example-namespace* is the non-reserved namespace and *examplebundle123* is the customCABundle value.

```
$ oc get configmap odh-trusted-ca-bundle -n example-namespace -o yaml | grep
examplebundle123
```

## 5.3. REMOVING A CA BUNDLE

You can remove a Certificate Authority (CA) bundle from all non-reserved namespaces in OpenShift AI. This process changes the default configuration and disables the creation of the **odh-trusted-ca-bundle** configuration file (ConfigMap), as described in *Understanding certificates in OpenShift AI*.



### NOTE

The **odh-trusted-ca-bundle** ConfigMaps are only deleted from namespaces when you set the **managementState** of **trustedCABundle** to **Removed**; deleting the DSC Initialization does not delete the ConfigMaps.

To remove a CA bundle from a single namespace only, see *Removing a CA bundle from a namespace*.

### Prerequisites

- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You installed the OpenShift command line interface (**oc**) as described in [Get Started with the CLI](#).

### Procedure

1. In the OpenShift Container Platform web console, click **Operators** → **Installed Operators** and then click the Red Hat OpenShift AI Operator.
2. Click the **DSC Initialization** tab.
3. Click the **default-dsci** object.
4. Click the **YAML** tab.
5. In the **spec** section, change the value of the **managementState** field for **trustedCABundle** to **Removed**:

```
spec:
  trustedCABundle:
    managementState: Removed
```

6. Click **Save**.

### Verification

- Run the following command to verify that the **odh-trusted-ca-bundle** ConfigMap has been removed from all namespaces:

```
$ oc get configmaps --all-namespaces | grep odh-trusted-ca-bundle
```

The command should not return any ConfigMaps.

## 5.4. REMOVING A CA BUNDLE FROM A NAMESPACE

You can remove a custom Certificate Authority (CA) bundle from individual namespaces in OpenShift AI. This process disables the creation of the **odh-trusted-ca-bundle** configuration file (ConfigMap) for the specified namespace only.

To remove a certificate bundle from all namespaces, see *Removing a CA bundle*.

### Prerequisites

- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You installed the OpenShift command line interface (**oc**) as described in [Get Started with the CLI](#).

### Procedure

- Run the following command to remove a CA bundle from a namespace. In the following command, *example-namespace* is the non-reserved namespace.

```
$ oc annotate ns example-namespace security.opendatahub.io/inject-trusted-ca-bundle=false
```

### Verification

- Run the following command to verify that the CA bundle has been removed from the namespace. In the following command, *example-namespace* is the non-reserved namespace.

```
$ oc get configmap odh-trusted-ca-bundle -n example-namespace
```

The command should return **configmaps "odh-trusted-ca-bundle" not found**.

## 5.5. MANAGING CERTIFICATES

After installing OpenShift AI, the Red Hat OpenShift AI Operator creates the **odh-trusted-ca-bundle** configuration file (ConfigMap) that contains the trusted CA bundle and adds it to all new and existing non-reserved namespaces in the cluster. By default, the Red Hat OpenShift AI Operator manages the **odh-trusted-ca-bundle** ConfigMap and automatically updates it if any changes are made to the CA bundle. You can choose to manage the **odh-trusted-ca-bundle** ConfigMap instead of allowing the Red Hat OpenShift AI Operator to manage it.

### Prerequisites

- You have cluster administrator privileges for your OpenShift Container Platform cluster.

### Procedure

1. In the OpenShift Container Platform web console, click **Operators** → **Installed Operators** and then click the **Red Hat OpenShift AI Operator**.
2. Click the **DSC Initialization** tab.
3. Click the **default-dsci** object.
4. Click the **YAML** tab.

- In the **spec** section, change the value of the **managementState** field for **trustedCABundle** to **Unmanaged**, as shown:

```
spec:
  trustedCABundle:
    managementState: Unmanaged
```

- Click **Save**.  
Note that changing the **managementState** from **Managed** to **Unmanaged** does not remove the **odh-trusted-ca-bundle** ConfigMap, but the ConfigMap is not updated if you make changes to the **customCABundle** field.

## Verification

- In the **spec** section, set or change the value of the **customCABundle** field for **trustedCABundle**, for example:

```
spec:
  trustedCABundle:
    managementState: Unmanaged
    customCABundle: example123
```

- Click **Save**.
- Click **Workloads → ConfigMaps**.
- Select a project from the project list.
- Click the **odh-trusted-ca-bundle** ConfigMap.
- Click the **YAML** tab and verify that the value of the **customCABundle** field did not update.

## 5.6. USING SELF-SIGNED CERTIFICATES WITH OPENSIFT AI COMPONENTS

Some OpenShift AI components have additional options or required configuration for self-signed certificates.

### 5.6.1. Using certificates with data science pipelines

If you want to use self-signed certificates, you have added them to a central Certificate Authority (CA) bundle as described in [Working with certificates](#) (for disconnected environments, see [Working with certificates](#)).

No additional configuration is necessary to use those certificates with data science pipelines.

#### 5.6.1.1. Providing a CA bundle only for data science pipelines

Perform the following steps to provide a Certificate Authority (CA) bundle just for data science pipelines.

#### Procedure

- Log in to OpenShift Container Platform.

- From **Workloads** → **ConfigMaps**, create a ConfigMap with the required bundle in the same data science project or namespace as the target data science pipeline:

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: custom-ca-bundle
data:
  ca-bundle.crt: |
    # contents of ca-bundle.crt
```

- Add the following snippet to the **.spec.apiserver.caBundle** field of the underlying Data Science Pipelines Application (DSPA):

```
apiVersion: datasciencepipelinesapplications.opendatahub.io/v1alpha1
kind: DataSciencePipelinesApplication
metadata:
  name: data-science-dspa
spec:
  ...
  apiServer:
  ...
  cABundle:
    configMapName: custom-ca-bundle
    configMapKey: ca-bundle.crt
```

The pipeline server pod redeploys with the updated bundle and uses it in the newly created pipeline pods.

## Verification

Perform the following steps to confirm that your CA bundle was successfully mounted.

- Log in to the OpenShift Container Platform console.
- Go to the OpenShift Container Platform project that corresponds to the data science project.
- Click the **Pods** tab.
- Click the pipeline server pod with the **ds-pipeline-dspa-`<hash>`** prefix.
- Click **Terminal**.
- Enter **cat /dsp-custom-certs/dsp-ca.crt**.
- Verify that your CA bundle is present within this file.

You can also confirm that your CA bundle was successfully mounted by using the CLI:

- In a terminal window, log in to the OpenShift cluster where OpenShift AI is deployed.

```
oc login
```

- Set the **dspa** value:

```
dspa=dspa
```

3. Set the **dsProject** value, replacing **\$YOUR\_DS\_PROJECT** with the name of your data science project:

```
dsProject=$YOUR_DS_PROJECT
```

4. Set the **pod** value:

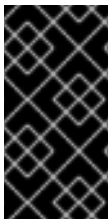
```
pod=$(oc get pod -n ${dsProject} -l app=ds-pipeline-${dspa} --no-headers | awk '{print $1}')
```

5. Display the contents of the **/dsp-custom-certs/dsp-ca.crt** file:

```
oc -n ${dsProject} exec $pod -- cat /dsp-custom-certs/dsp-ca.crt
```

6. Verify that your CA bundle is present within this file.

## 5.6.2. Using certificates with workbenches



### IMPORTANT

Self-signed certificates apply to workbenches that you create after configuring self-signed certificates centrally as described in [Working with certificates](#) (for disconnected environments, see [Working with certificates](#)). There is no change to workbenches that you created before configuring self-signed certificates.

### 5.6.2.1. Creating data science pipelines with Elyra and self-signed certificates

To create pipelines using a workbench that contains the Elyra extension and which uses self-signed certificates, see the [Workbench workaround for executing a pipeline using Elyra in a disconnected environment](#) knowledgebase article.

## CHAPTER 6. ENABLING GPU SUPPORT IN OPENSIFT AI

Optionally, to ensure that your data scientists can use compute-heavy workloads in their models, you can enable graphics processing units (GPUs) in OpenShift AI.

### Prerequisites

- You have logged in to your OpenShift Container Platform cluster.
- You have the **cluster-admin** role in your OpenShift Container Platform cluster.

### Procedure

1. To enable GPU support on an OpenShift cluster in a disconnected or airgapped environment, follow the instructions here: [Deploy GPU Operators in a disconnected or airgapped environment](#) in the NVIDIA documentation.
2. Delete the **migration-gpu-status** ConfigMap.
  - a. In the OpenShift Container Platform web console, switch to the **Administrator** perspective.
  - b. Set the **Project** to **All Projects** or **redhat-ods-applications** to ensure you can see the appropriate ConfigMap.
  - c. Search for the **migration-gpu-status** ConfigMap.
  - d. Click the action menu ( **:** ) and select **Delete ConfigMap** from the list.  
The **Delete ConfigMap** dialog appears.
  - e. Inspect the dialog and confirm that you are deleting the correct ConfigMap.
  - f. Click **Delete**.
3. Restart the dashboard replicaset.
  - a. In the OpenShift Container Platform web console, switch to the **Administrator** perspective.
  - b. Click **Workloads** → **Deployments**.
  - c. Set the **Project** to **All Projects** or **redhat-ods-applications** to ensure you can see the appropriate deployment.
  - d. Search for the **rhods-dashboard** deployment.
  - e. Click the action menu ( **:** ) and select **Restart Rollout** from the list.
  - f. Wait until the **Status** column indicates that all pods in the rollout have fully restarted.

### Verification

- The NVIDIA GPU Operator appears on the **Operators** → **Installed Operators** page in the OpenShift Container Platform web console.
- The reset **migration-gpu-status** instance is present in the **Instances** tab on the **AcceleratorProfile** custom resource definition (CRD) details page.



After installing the NVIDIA GPU Operator, create an accelerator profile as described in [Working with accelerator profiles](#).


## CHAPTER 7. ACCESSING THE DASHBOARD

After you have installed OpenShift AI and added users, you can access the URL for your OpenShift AI console and share the URL with the users to let them log in and work on their models.

### Prerequisites

- You have installed OpenShift AI on your OpenShift Container Platform cluster.
- You have added at least one user to the user group for OpenShift AI.

### Procedure

1. Log in to OpenShift Container Platform web console.
2. Click the application launcher (  ).
3. Right-click on **Red Hat OpenShift AI** and copy the URL for your OpenShift AI instance.
4. Provide this instance URL to your data scientists to let them log in to OpenShift AI.

### Verification

- Confirm that you and your users can log in to OpenShift AI by using the instance URL.

### Additional resources

- [Logging in to OpenShift AI](#)
- [Adding users](#)

# CHAPTER 8. UNINSTALLING RED HAT OPENSIFT AI SELF-MANAGED

This section shows how to use the OpenShift command-line interface (CLI) to uninstall the Red Hat OpenShift AI Operator and any OpenShift AI components installed and managed by the Operator.



## NOTE

Using the CLI is the recommended way to uninstall the Operator. Depending on your version of OpenShift Container Platform, using the web console to perform the uninstallation might not prompt you to uninstall all associated components. This could leave you unclear about the final state of your cluster.

## 8.1. UNDERSTANDING THE UNINSTALLATION PROCESS

Installing Red Hat OpenShift AI created several custom resource instances on your OpenShift Container Platform cluster for various components of OpenShift AI. After installation, users likely created several additional resources while using OpenShift AI. Uninstalling OpenShift AI removes the resources that were created by the Operator, but retains the resources created by users to prevent inadvertently deleting information you might want.

### What is deleted

Uninstalling OpenShift AI removes the following resources from your OpenShift Container Platform cluster:

- **DataScienceCluster** custom resource instance
- **DSCInitialization** custom resource instance
- **FeatureTracker** custom resource instances created during or after installation
- **ServiceMesh** custom resource instance created by the Operator during or after installation
- **KNativeServing** custom resource instance created by the Operator during or after installation
- **redhat-ods-applications**, **redhat-ods-monitoring**, and **rhods-notebooks** namespaces created by the Operator
- Workloads in the **rhods-notebooks** namespace
- **Subscription**, **ClusterServiceVersion**, and **InstallPlan** objects
- **KfDef** object (version 1 Operator only)

### What might remain

Uninstalling OpenShift AI retains the following resources in your OpenShift Container Platform cluster:

- Data science projects created by users
- Custom resource instances created by users
- Custom resource definitions (CRDs) created by users or by the Operator

While these resources might still remain in your OpenShift Container Platform cluster, they are not

functional. After uninstalling, Red Hat recommends that you review the data science projects and custom resources in your OpenShift Container Platform cluster and delete anything no longer in use to prevent potential issues, such as pipelines that cannot run, notebooks that cannot be undeployed, or models that cannot be undeployed.

### Additional resources

[Operator Lifecycle Manager \(OLM\) uninstall documentation](#)

## 8.2. UNINSTALLING OPENSIFT AI SELF-MANAGED BY USING THE CLI

The following procedure shows how to use the OpenShift command-line interface (CLI) to uninstall the Red Hat OpenShift AI Operator and any OpenShift AI components installed and managed by the Operator.

### Prerequisites

- You have cluster administrator privileges for your OpenShift Container Platform cluster.
- You have downloaded and installed the OpenShift command-line interface (CLI). See [Installing the OpenShift CLI](#).
- You have backed up the persistent disks or volumes used by your persistent volume claims (PVCs).

### Procedure

1. Open a new terminal window.
2. In the OpenShift command-line interface (CLI), log in to your OpenShift Container Platform cluster as a cluster administrator, as shown in the following example:

```
$ oc login <openshift_cluster_url> -u system:admin
```

3. Create a **ConfigMap** object for deletion of the Red Hat OpenShift AI Operator.

```
$ oc create configmap delete-self-managed-odh -n redhat-ods-operator
```

4. To delete the **rhods-operator**, set the **addon-managed-odh-delete** label to **true**.

```
$ oc label configmap/delete-self-managed-odh api.openshift.com/addon-managed-odh-delete=true -n redhat-ods-operator
```

5. When all objects associated with the Operator are removed, delete the **redhat-ods-operator** project.

- a. Set an environment variable for the **redhat-ods-applications** project.

```
$ PROJECT_NAME=redhat-ods-applications
```

- b. Wait until the **redhat-ods-applications** project has been deleted.

```
$ while oc get project $PROJECT_NAME &> /dev/null; do
```

```
echo "The $PROJECT_NAME project still exists"
sleep 1
done
echo "The $PROJECT_NAME project no longer exists"
```

When the **redhat-ods-applications** project has been deleted, you see the following output.

```
The redhat-ods-applications project no longer exists
```

- c. When the **redhat-ods-applications** project has been deleted, delete the **redhat-ods-operator** project.

```
$ oc delete namespace redhat-ods-operator
```

## Verification

- Confirm that the **rhods-operator** subscription no longer exists.

```
$ oc get subscriptions --all-namespaces | grep rhods-operator
```

- Confirm that the following projects no longer exist.

- **redhat-ods-applications**
- **redhat-ods-monitoring**
- **redhat-ods-operator**
- **rhods-notebooks**

```
$ oc get namespaces | grep -e redhat-ods* -e rhods*
```



### NOTE

The **rhods-notebooks** project was created only if you installed the workbenches component of OpenShift AI. See [Installing and managing Red Hat OpenShift AI components](#).