



Red Hat OpenShift Data Foundation 4.15

Deploying OpenShift Data Foundation using IBM Z

Instructions on deploying Red Hat OpenShift Data Foundation to use local storage on
IBM Z

Red Hat OpenShift Data Foundation 4.15 Deploying OpenShift Data Foundation using IBM Z

Instructions on deploying Red Hat OpenShift Data Foundation to use local storage on IBM Z

Legal Notice

Copyright © 2024 Red Hat, Inc.

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

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

Abstract

Read this document for instructions about how to install Red Hat OpenShift Data Foundation to use local storage on IBM Z. While this document refers only to IBM Z, all information in it also applies to IBM(R) LinuxONE.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	3
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	4
PREFACE	5
CHAPTER 1. PREPARING TO DEPLOY OPENSIFT DATA FOUNDATION	6
1.1. REQUIREMENTS FOR INSTALLING OPENSIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES	6
1.2. ENABLING CLUSTER-WIDE ENCRYPTION WITH KMS USING THE TOKEN AUTHENTICATION METHOD	6
CHAPTER 2. DEPLOY OPENSIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES	8
2.1. INSTALLING RED HAT OPENSIFT DATA FOUNDATION OPERATOR	8
2.2. INSTALLING LOCAL STORAGE OPERATOR	9
2.3. FINDING AVAILABLE STORAGE DEVICES (OPTIONAL)	10
2.4. CREATING OPENSIFT DATA FOUNDATION CLUSTER ON IBM Z	11
CHAPTER 3. VERIFYING OPENSIFT DATA FOUNDATION DEPLOYMENT FOR INTERNAL-ATTACHED DEVICES MODE	15
3.1. VERIFYING THE STATE OF THE PODS	15
3.2. VERIFYING THE OPENSIFT DATA FOUNDATION CLUSTER IS HEALTHY	17
3.3. VERIFYING THE MULTICLOUD OBJECT GATEWAY IS HEALTHY	17
3.4. VERIFYING THAT THE SPECIFIC STORAGE CLASSES EXIST	17
CHAPTER 4. VIEW OPENSIFT DATA FOUNDATION TOPOLOGY	19
CHAPTER 5. DEPLOYING STANDALONE MULTICLOUD OBJECT GATEWAY	20
PART I. INSTALLING LOCAL STORAGE OPERATOR	21
1. INSTALLING RED HAT OPENSIFT DATA FOUNDATION OPERATOR	21
2. CREATING STANDALONE MULTICLOUD OBJECT GATEWAY ON IBM Z	23
CHAPTER 6. VIEW OPENSIFT DATA FOUNDATION TOPOLOGY	27
CHAPTER 7. UNINSTALLING OPENSIFT DATA FOUNDATION	28
7.1. Uninstalling OpenShift Data Foundation in Internal-attached devices mode	28
7.1.1. Removing local storage operator configurations	33
7.2. Removing monitoring stack from OpenShift Data Foundation	37
7.3. Removing OpenShift Container Platform registry from OpenShift Data Foundation	40
7.4. Removing the cluster logging operator from OpenShift Data Foundation	41

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Do let us know how we can make it better.

To give feedback, create a Bugzilla ticket:

1. Go to the [Bugzilla](#) website.
2. In the **Component** section, choose **documentation**.
3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
4. Click **Submit Bug**.

PREFACE

Red Hat OpenShift Data Foundation supports deployment on existing Red Hat OpenShift Container Platform (RHOCP) IBM Z clusters in connected or disconnected environments along with out-of-the-box support for proxy environments.



NOTE

See [Planning your deployment](#) and [Preparing to deploy OpenShift Data Foundation](#) for more information about deployment requirements.

To deploy OpenShift Data Foundation, follow the appropriate deployment process for your environment:

- Internal Attached Devices mode
 - [Deploy using local storage devices](#)
- [External mode](#)

CHAPTER 1. PREPARING TO DEPLOY OPENSIFT DATA FOUNDATION

When you deploy OpenShift Data Foundation on OpenShift Container Platform using local storage devices, you can create internal cluster resources. This approach internally provisions base services and all applications can access additional storage classes.

Before you begin the deployment of Red Hat OpenShift Data Foundation using local storage, ensure that your resource requirements are met. See [requirements for installing OpenShift Data Foundation using local storage devices](#).

On the external key management system (KMS),

- When the Token authentication method is selected for encryption then refer to [Enabling cluster-wide encryption with the Token authentication using KMS](#).
- Ensure that you are using signed certificates on your Vault servers.

After you have addressed the above, follow these steps in the order given:

1. [Install the Red Hat OpenShift Data Foundation Operator](#) .
2. [Install Local Storage Operator](#) .
3. [Find the available storage devices](#) .
4. [Create the OpenShift Data Foundation cluster service on IBM Z](#) .

1.1. REQUIREMENTS FOR INSTALLING OPENSIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES

Node requirements

The cluster must consist of at least three OpenShift Container Platform worker or infrastructure nodes with locally attached-storage devices on each of them.

- Each of the three selected nodes must have at least one raw block device available. OpenShift Data Foundation uses the one or more available raw block devices.
- The devices you use must be empty, the disks must not include Physical Volumes (PVs), Volume Groups (VGs), or Logical Volumes (LVs) remaining on the disk.

For more information, see the *Resource requirements* section in the [Planning guide](#).

1.2. ENABLING CLUSTER-WIDE ENCRYPTION WITH KMS USING THE TOKEN AUTHENTICATION METHOD

You can enable the key value backend path and policy in the vault for token authentication.

Prerequisites

- Administrator access to the vault.

- A valid Red Hat OpenShift Data Foundation Advanced subscription. For more information, see the [knowledgebase article on OpenShift Data Foundation subscriptions](#).
- Carefully, select a unique path name as the backend **path** that follows the naming convention since you cannot change it later.

Procedure

1. Enable the Key/Value (KV) backend path in the vault.

For vault KV secret engine API, version 1:

```
$ vault secrets enable -path=odf kv
```

For vault KV secret engine API, version 2:

```
$ vault secrets enable -path=odf kv-v2
```

2. Create a policy to restrict the users to perform a write or delete operation on the secret:

```
echo '  
path "odf/*" {  
  capabilities = ["create", "read", "update", "delete", "list"]  
}  
path "sys/mounts" {  
  capabilities = ["read"]  
}' | vault policy write odf -
```

3. Create a token that matches the above policy:

```
$ vault token create -policy=odf -format json
```

CHAPTER 2. DEPLOY OPENSHIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES

Deploying OpenShift Data Foundation on OpenShift Container Platform using local storage devices provides you with the option to create internal cluster resources. Follow this deployment method to use local storage to back persistent volumes for your OpenShift Container Platform applications.

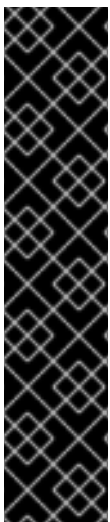
Use this section to deploy OpenShift Data Foundation on IBM Z infrastructure where OpenShift Container Platform is already installed.

2.1. INSTALLING RED HAT OPENSHIFT DATA FOUNDATION OPERATOR

You can install Red Hat OpenShift Data Foundation Operator using the Red Hat OpenShift Container Platform Operator Hub.

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** and operator installation permissions.
- You must have at least three worker or infrastructure nodes in the Red Hat OpenShift Container Platform cluster.
- For additional resource requirements, see the [Planning your deployment](#) guide.



IMPORTANT

- When you need to override the cluster-wide default node selector for OpenShift Data Foundation, you can use the following command to specify a blank node selector for the **openshift-storage** namespace (create **openshift-storage** namespace in this case):


```
$ oc annotate namespace openshift-storage openshift.io/node-selector=
```
- Taint a node as **infra** to ensure only Red Hat OpenShift Data Foundation resources are scheduled on that node. This helps you save on subscription costs. For more information, see the *How to use dedicated worker nodes for Red Hat OpenShift Data Foundation* section in the [Managing and Allocating Storage Resources](#) guide.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators** → **OperatorHub**.
3. Scroll or type **OpenShift Data Foundation** into the **Filter by keyword** box to find the **OpenShift Data Foundation Operator**.
4. Click **Install**.
5. Set the following options on the **Install Operator** page:

- a. Update Channel as **stable-4.15**.
- b. Installation Mode as **A specific namespace on the cluster**
- c. Installed Namespace as **Operator recommended namespace openshift-storage**. If Namespace **openshift-storage** does not exist, it is created during the operator installation.
- d. Select Approval Strategy as **Automatic** or **Manual**.
If you select **Automatic** updates, then the Operator Lifecycle Manager (OLM) automatically upgrades the running instance of your Operator without any intervention.

If you select **Manual** updates, then the OLM creates an update request. As a cluster administrator, you must then manually approve that update request to update the Operator to a newer version.
- e. Ensure that the **Enable** option is selected for the **Console plugin**.
- f. Click **Install**.

Verification steps

- After the operator is successfully installed, a pop-up with a message, **Web console update is available** appears on the user interface. Click **Refresh web console** from this pop-up for the console changes to reflect.
- In the Web Console:
 - Navigate to Installed Operators and verify that the **OpenShift Data Foundation** Operator shows a green tick indicating successful installation.
 - Navigate to **Storage** and verify if the **Data Foundation** dashboard is available.

2.2. INSTALLING LOCAL STORAGE OPERATOR

Install the Local Storage Operator from the Operator Hub before creating Red Hat OpenShift Data Foundation clusters on local storage devices.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators → OperatorHub**.
3. Type **local storage** in the **Filter by keyword** box to find the **Local Storage Operator** from the list of operators, and click on it.
4. Set the following options on the **Install Operator** page:
 - a. Update channel as **stable**.
 - b. Installation mode as **A specific namespace on the cluster**
 - c. Installed Namespace as **Operator recommended namespace openshift-local-storage**.
 - d. Update approval as **Automatic**.
5. Click **Install**.

Verification steps

- Verify that the Local Storage Operator shows a green tick indicating successful installation.

2.3. FINDING AVAILABLE STORAGE DEVICES (OPTIONAL)

This step is additional information and can be skipped as the disks are automatically discovered during storage cluster creation. Use this procedure to identify the device names for each of the three or more worker nodes that you have labeled with the OpenShift Data Foundation label **cluster.ocs.openshift.io/openshift-storage=** before creating Persistent Volumes (PV) for IBM Z.

Procedure

1. List and verify the name of the worker nodes with the OpenShift Data Foundation label.

```
$ oc get nodes -l=cluster.ocs.openshift.io/openshift-storage=
```

Example output:

```
NAME          STATUS  ROLES  AGE   VERSION
bmworker01   Ready  worker 6h45m v1.16.2
bmworker02   Ready  worker 6h45m v1.16.2
bmworker03   Ready  worker 6h45m v1.16.2
```

2. Log in to each worker node that is used for OpenShift Data Foundation resources and find the unique **by-id** device name for each available raw block device.

```
$ oc debug node/<node name>
```

Example output:

```
$ oc debug node/bmworker01
Starting pod/bmworker01-debug ...
To use host binaries, run `chroot /host`
Pod IP: 10.0.135.71
If you don't see a command prompt, try pressing enter.
sh-4.2# chroot /host
sh-4.4# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0                7:0    0  500G  0 loop
sda                  8:0    0  120G  0 disk
|-sda1                8:1    0   384M  0 part /boot
`-sda4                8:4    0  119.6G  0 part
`-coreos-luks-root-nocrypt 253:0  0  119.6G  0 dm  /sysroot
sdb                  8:16   0  500G  0 disk
```

In this example, for **bmworker01**, the available local device is **sdb**.

3. Identify the unique ID for each of the devices selected in Step 2.

```
sh-4.4#ls -l /dev/disk/by-id/ | grep sdb
lrwxrwxrwx. 1 root root 9 Feb 3 16:49 scsi-360050763808104bc2800000000000259 ->
../sdb
```

```
lrwxrwxrwx. 1 root root 9 Feb 3 16:49 scsi-SIBM_2145_00e020412f0aXX00 -> ../../sdb
lrwxrwxrwx. 1 root root 9 Feb 3 16:49 scsi-0x60050763808104bc2800000000000259 ->
../../sdb
```

In the above example, the ID for the local device **sdb**

```
scsi-0x60050763808104bc2800000000000259
```

4. Repeat the above step to identify the device ID for all the other nodes that have the storage devices to be used by OpenShift Data Foundation. See this [Knowledge Base article](#) for more details.

2.4. CREATING OPENSIFT DATA FOUNDATION CLUSTER ON IBM Z

Use this procedure to create an OpenShift Data Foundation cluster on IBM Z.

Prerequisites

- Ensure that all the requirements in the [Requirements for installing OpenShift Data Foundation using local storage devices](#) section are met.
- You must have at least three worker nodes with the same storage type and size attached to each node (for example, 200 GB) to use local storage devices on IBM Z or IBM® LinuxONE.

Procedure

1. In the OpenShift Web Console, click **Operators** → **Installed Operators** to view all the installed operators.
Ensure that the **Project** selected is **openshift-storage**.
2. Click on the **OpenShift Data Foundation** operator and then click **Create StorageSystem**.
3. In the Backing storage page, perform the following:
 - a. Select the **Create a new StorageClass using the local storage devices for Backing storage type** option.
 - b. Select **Full Deployment** for the **Deployment type** option.
 - c. Click Next.



IMPORTANT

You are prompted to install the Local Storage Operator if it is not already installed. Click **Install**, and follow the procedure as described in [Installing Local Storage Operator](#).

4. In the **Create local volume set** page, provide the following information:
 - a. Enter a name for the **LocalVolumeSet** and the **StorageClass**.
By default, the local volume set name appears for the storage class name. You can change the name.
 - b. Choose one of the following:

- **Disks on all nodes**
Uses the available disks that match the selected filters on all the nodes.
- **Disks on selected nodes**
Uses the available disks that match the selected filters only on the selected nodes.



IMPORTANT

- The flexible scaling feature is enabled only when the storage cluster that you created with three or more nodes are spread across fewer than the minimum requirement of three availability zones.
For information about flexible scaling, see [knowledgebase article on Scaling OpenShift Data Foundation cluster using YAML when flexible scaling is enabled](#).
- Flexible scaling features get enabled at the time of deployment and can not be enabled or disabled later on.
- If the nodes selected do not match the OpenShift Data Foundation cluster requirement of an aggregated 30 CPUs and 72 GiB of RAM, a minimal cluster is deployed.
For minimum starting node requirements, see the [Resource requirements](#) section in the *Planning* guide.

- From the available list of **Disk Type**, select **SSD/NVME**.
- Expand the **Advanced** section and set the following options:

Volume Mode	Block is selected by default.
Device Type	Select one or more device type from the dropdown list. Note: For a multi-path device, select the Mpath option from the dropdown exclusively.
Disk Size	Set a minimum size of 100GB for the device and maximum available size of the device that needs to be included.
Maximum Disks Limit	This indicates the maximum number of PVs that can be created on a node. If this field is left empty, then PVs are created for all the available disks on the matching nodes.

- Click **Next**.
A pop-up to confirm the creation of LocalVolumeSet is displayed.
 - Click **Yes** to continue.
- In the **Capacity and nodes** page, configure the following:
 - Available raw capacity** is populated with the capacity value based on all the attached disks associated with the storage class. This takes some time to show up. The **Selected nodes** list shows the nodes based on the storage class.
 - You can check the box to select **Taint** nodes.

- c. Click **Next**.
6. Optional: In the **Security and network** page, configure the following based on your requirement:
 - a. To enable encryption, select **Enable data encryption for block and file storage**
 - b. Choose one or both of the following **Encryption level**:
 - **Cluster-wide encryption**
Encrypts the entire cluster (block and file).
 - **StorageClass encryption**
Creates encrypted persistent volume (block only) using encryption enabled storage class.
 - c. Select **Connect to an external key management service** checkbox. This is optional for cluster-wide encryption.
 - i. **Key Management Service Provider** is set to **Vault** by default.
 - ii. Enter Vault **Service Name**, host **Address** of Vault server ('https://<hostname or ip>'), **Port** number and **Token**.
 - iii. Expand **Advanced Settings** to enter additional settings and certificate details based on your Vault configuration:
 - A. Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
 - B. Optional: Enter **TLS Server Name** and **Vault Enterprise Namespace**
 - C. Upload the respective PEM encoded certificate file to provide **CA Certificate**, **Client Certificate** and **Client Private Key**.
 - D. Click **Save**.
 - d. Select **Default (SDN)** as Multus is not yet supported on OpenShift Data Foundation on IBM Z.
 - e. Click **Next**.
7. In the **Data Protection** page, if you are configuring Regional-DR solution for Openshift Data Foundation then select the **Prepare cluster for disaster recovery (Regional-DR only)** checkbox, else click **Next**.
8. In the Review and create page::
 - a. Review the configuration details. To modify any configuration settings, click **Back** to go back to the previous configuration page.
 - b. Click **Create StorageSystem**.

Verification steps

- To verify the final Status of the installed storage cluster:

- a. In the OpenShift Web Console, navigate to **Installed Operators** → **OpenShift Data Foundation** → **Storage System** → **ocs-storagecluster-storagesystem** → **Resources**.
 - b. Verify that **Status** of **StorageCluster** is **Ready** and has a green tick mark next to it.
- To verify if flexible scaling is enabled on your storage cluster, perform the following steps:
 1. In the OpenShift Web Console, navigate to **Installed Operators** → **OpenShift Data Foundation** → **Storage System** → **ocs-storagecluster-storagesystem** → **Resources** → **ocs-storagecluster**.
 2. In the YAML tab, search for the keys **flexibleScaling** in **spec** section and **failureDomain** in **status** section. If **flexible scaling** is true and **failureDomain** is set to host, flexible scaling feature is enabled.

```
spec:  
  flexibleScaling: true  
  [...]  
status:  
  failureDomain: host
```

- To verify that all components for OpenShift Data Foundation are successfully installed, see [Verifying your OpenShift Data Foundation deployment](#).

Additional resources

- To expand the capacity of the initial cluster, see the [Scaling Storage](#) guide.

CHAPTER 3. VERIFYING OPENSIFT DATA FOUNDATION DEPLOYMENT FOR INTERNAL-ATTACHED DEVICES MODE

Use this section to verify that OpenShift Data Foundation is deployed correctly.

3.1. VERIFYING THE STATE OF THE PODS

Procedure

1. Click **Workloads** → **Pods** from the OpenShift Web Console.
2. Select **openshift-storage** from the **Project** drop-down list.



NOTE

If the **Show default projects** option is disabled, use the toggle button to list all the default projects.

For more information on the expected number of pods for each component and how it varies depending on the number of nodes, see [Table 3.1, “Pods corresponding to OpenShift Data Foundation cluster”](#).

3. Set filter for Running and Completed pods to verify that the following pods are in **Running** and **Completed** state:

Table 3.1. Pods corresponding to OpenShift Data Foundation cluster

Component	Corresponding pods
OpenShift Data Foundation Operator	<ul style="list-style-type: none"> ● ocs-operator-* (1 pod on any storage node) ● ocs-metrics-exporter-* (1 pod on any storage node) ● odf-operator-controller-manager-* (1 pod on any storage node) ● csi-addons-controller-manager-* (1 pod on any storage node) ● odf-console-* (1 pod on any storage node)
Rook-ceph Operator	<p>rook-ceph-operator-*</p> <p>(1 pod on any storage node)</p>

Component	Corresponding pods
Multicloud Object Gateway	<ul style="list-style-type: none"> ● noobaa-operator-* (1 pod on any storage node) ● noobaa-core-* (1 pod on any storage node) ● noobaa-db-pg-* (1 pod on any storage node) ● noobaa-endpoint-* (1 pod on any storage node)
MON	<p>rook-ceph-mon-*</p> <p>(3 pods distributed across storage nodes)</p>
MGR	<p>rook-ceph-mgr-*</p> <p>(1 pod on any storage node)</p>
MDS	<p>rook-ceph-mds-ocs-storagecluster-cephfilesystem-*</p> <p>(2 pods distributed across storage nodes)</p>
RGW	<p>rook-ceph-rgw-ocs-storagecluster-cephobjectstore-* (1 pod on any storage node)</p>
CSI	<ul style="list-style-type: none"> ● cephfs <ul style="list-style-type: none"> ○ csi-cephfsplugin-* (1 pod on each storage node) ○ csi-cephfsplugin-provisioner-* (2 pods distributed across storage nodes) ● rbd <ul style="list-style-type: none"> ○ csi-rbdplugin-* (1 pod on each storage node) ○ csi-rbdplugin-provisioner-* (2 pods distributed across storage nodes)
rook-ceph-crashcollector	<p>rook-ceph-crashcollector-*</p> <p>(1 pod on each storage node)</p>

Component	Corresponding pods
OSD	<ul style="list-style-type: none"> ● rook-ceph-osd-* (1 pod for each device) ● rook-ceph-osd-prepare-ocs-deviceset-* (1 pod for each device)

3.2. VERIFYING THE OPENSIFT DATA FOUNDATION CLUSTER IS HEALTHY

Procedure

1. In the OpenShift Web Console, click **Storage** → **Data Foundation**.
2. Click the **Storage Systems** tab and then click on **ocs-storagecluster-storagesystem**.
3. In the **Status card** of Block and File dashboard under Overview tab, verify that both *Storage Cluster* and *Data Resiliency* has a green tick mark.
4. In the **Details card**, verify that the cluster information is displayed.

For more information on the health of the OpenShift Data Foundation cluster using the Block and File dashboard, see [Monitoring OpenShift Data Foundation](#).

3.3. VERIFYING THE MULTICLOUD OBJECT GATEWAY IS HEALTHY

Procedure

1. In the OpenShift Web Console, click **Storage** → **Data Foundation**.
2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.
 - a. In the **Status card** of the **Object** tab, verify that both *Object Service* and *Data Resiliency* have a green tick.
 - b. In the **Details** card, verify that the MCG information is displayed.

For more information on the health of the OpenShift Data Foundation cluster using the object service dashboard, see [Monitoring OpenShift Data Foundation](#).

3.4. VERIFYING THAT THE SPECIFIC STORAGE CLASSES EXIST

Procedure

1. Click **Storage** → **Storage Classes** from the left pane of the OpenShift Web Console.
2. Verify that the following storage classes are created with the OpenShift Data Foundation cluster creation:
 - **ocs-storagecluster-ceph-rbd**

- **ocs-storagecluster-cephfs**
- **openshift-storage.noobaa.io**
- **ocs-storagecluster-ceph-rgw**

CHAPTER 4. VIEW OPENSIFT DATA FOUNDATION TOPOLOGY

The topology shows the mapped visualization of the OpenShift Data Foundation storage cluster at various abstraction levels and also lets you to interact with these layers. The view also shows how the various elements compose the Storage cluster altogether.

Procedure

1. On the OpenShift Web Console, navigate to **Storage → Data Foundation → Topology**.
The view shows the storage cluster and the zones inside it. You can see the nodes depicted by circular entities within the zones, which are indicated by dotted lines. The label of each item or resource contains basic information such as status and health or indication for alerts.
2. Choose a node to view node details on the right-hand panel. You can also access resources or deployments within a node by clicking on the search/preview decorator icon.
3. To view deployment details
 - a. Click the preview decorator on a node. A modal window appears above the node that displays all of the deployments associated with that node along with their statuses.
 - b. Click the **Back to main view** button in the model's upper left corner to close and return to the previous view.
 - c. Select a specific deployment to see more information about it. All relevant data is shown in the side panel.
4. Click the **Resources** tab to view the pods information. This tab provides a deeper understanding of the problems and offers granularity that aids in better troubleshooting.
5. Click the pod links to view the pod information page on OpenShift Container Platform. The link opens in a new window.

CHAPTER 5. DEPLOYING STANDALONE MULTICLOUD OBJECT GATEWAY

Deploying only the Multicloud Object Gateway component with the OpenShift Data Foundation provides the flexibility in deployment and helps to reduce the resource consumption. Use this section to deploy only the standalone Multicloud Object Gateway component, which involves the following steps: *

- * Installing the Local Storage Operator.
- * Installing Red Hat OpenShift Data Foundation Operator *
- * Creating standalone Multicloud Object Gateway

:leveloffset: +2

PART I. INSTALLING LOCAL STORAGE OPERATOR

Use this procedure to install the Local Storage Operator from the Operator Hub before creating OpenShift Data Foundation clusters on local storage devices.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators** → **OperatorHub**.
3. Type **local storage** in the **Filter by keyword...** box to find the **Local Storage Operator** from the list of operators and select the same.
4. Set the following options on the **Install Operator** page:
 - a. Update channel as **stable**.
 - b. Installation Mode as **A specific namespace on the cluster**
 - c. Installed Namespace as **Operator recommended namespace openshift-local-storage**.
 - d. Approval Strategy as **Automatic**.
5. Click **Install**.

Verification steps

- Verify that the Local Storage Operator shows a green tick indicating successful installation.

1. INSTALLING RED HAT OPENSIFT DATA FOUNDATION OPERATOR

You can install Red Hat OpenShift Data Foundation Operator by using the Red Hat OpenShift Container Platform Operator Hub.

For information about the hardware and software requirements, see [Planning your deployment](#).

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** and Operator installation permissions.
- You must have at least three worker nodes in the Red Hat OpenShift Container Platform cluster.



IMPORTANT

- When you need to override the cluster-wide default node selector for OpenShift Data Foundation, you can use the following command in the command line interface to specify a blank node selector for the **openshift-storage** namespace (create **openshift-storage** namespace in this case):

```
$ oc annotate namespace openshift-storage openshift.io/node-selector=
```

- Taint a node as **infra** to ensure only Red Hat OpenShift Data Foundation resources are scheduled on that node. This helps you save on subscription costs. For more information, see [How to use dedicated worker nodes for Red Hat OpenShift Data Foundation](#) chapter in the *Managing and Allocating Storage Resources* guide.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators** → **OperatorHub**.
3. Scroll or type **OpenShift Data Foundation** into the **Filter by keyword** box to find the **OpenShift Data Foundation** Operator.
4. Click **Install**.
5. Set the following options on the **Install Operator** page:
 - a. Update Channel as **stable-4.15**.
 - b. Installation Mode as **A specific namespace on the cluster**
 - c. Installed Namespace as **Operator recommended namespace openshift-storage**. If Namespace **openshift-storage** does not exist, it is created during the operator installation.
6. Select **Approval Strategy** as **Automatic** or **Manual**.
If you select **Automatic** updates, then the Operator Lifecycle Manager (OLM) automatically upgrades the running instance of your Operator without any intervention.

If you select **Manual** updates, then the OLM creates an update request. As a cluster administrator, you must then manually approve that update request to update the Operator to a newer version.
7. Ensure that the **Enable** option is selected for the **Console plugin**.
8. Click **Install**.

Verification steps

- Verify that the **OpenShift Data Foundation** Operator shows a green tick indicating successful installation.
- After the operator is successfully installed, a pop-up with a message **Web console update is available** appears on the user interface. Click **Refresh web console** from this pop-up for the console changes to reflect.
 - In the Web Console, navigate to **Storage** and verify if **Data Foundation** is available.

2. CREATING STANDALONE MULTICLOUD OBJECT GATEWAY ON IBM Z

You can create only the standalone Multicloud Object Gateway component while deploying OpenShift Data Foundation.

Prerequisites

- Ensure that the OpenShift Data Foundation Operator is installed.
- (For deploying using local storage devices only) Ensure that Local Storage Operator is installed.

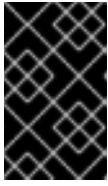
To identify storage devices on each node, see [Finding available storage devices](#).

Procedure

1. Log into the OpenShift Web Console.
2. In **openshift-local-storage** namespace, click **Operators** → **Installed Operators** to view the installed operators.
3. Click the **Local Storage** installed operator.
4. On the **Operator Details** page, click the **Local Volume** link.
5. Click **Create Local Volume**.
6. Click on **YAML view** for configuring Local Volume.
7. Define a **LocalVolume** custom resource for filesystem PVs using the following YAML.

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: localblock
  namespace: openshift-local-storage
spec:
  logLevel: Normal
  managementState: Managed
  nodeSelector:
    nodeSelectorTerms:
      - matchExpressions:
          - key: kubernetes.io/hostname
            operator: In
            values:
              - worker-0
              - worker-1
              - worker-2
  storageClassDevices:
    - devicePaths:
        - /dev/sda
      storageClassName: localblock
      volumeMode: Filesystem
```

The above definition selects **sda** local device from the **worker-0**, **worker-1** and **worker-2** nodes. The **localblock** storage class is created and persistent volumes are provisioned from **sda**.



IMPORTANT

Specify appropriate values of nodeSelector as per your environment. The device name should be same on all the worker nodes. You can also specify more than one devicePaths.

8. Click **Create**.
9. In the OpenShift Web Console, click **Operators** → **Installed Operators** to view all the installed operators.
Ensure that the **Project** selected is **openshift-storage**.
10. Click **OpenShift Data Foundation** operator and then click **Create StorageSystem**.
11. In the **Backing storage** page, select **Multicloud Object Gateway** for **Deployment type**.
12. Select the **Use an existing StorageClass** option for **Backing storage type**.
 - a. Select the **Storage Class** that you used while installing LocalVolume.
13. Click **Next**.
14. Optional: In the **Security** page, select the **Connect to an external key management service** checkbox. This is optional for cluster-wide encryption.
 - a. From the **Key Management Service Provider** drop down list, either select **Vault** or **Thales CipherTrust Manager (using KMIP)**. If you selected **Vault**, go to the next step. If you selected **Thales CipherTrust Manager (using KMIP)**, go to step iii.
 - b. Select an **Authentication Method**.

Using Token authentication method

- Enter a unique **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Token**.
- Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
 - Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
 - Optional: Enter **TLS Server Name** and **Vault Enterprise Namespace**
 - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate** and **Client Private Key**.
 - Click **Save** and skip to step iv.

Using Kubernetes authentication method

- Enter a unique Vault **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Role** name.

- Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
 - Enter the Key Value secret path in the **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
 - Optional: Enter **TLS Server Name** and **Authentication Path** if applicable.
 - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate**, and **Client Private Key**.
 - Click **Save** and skip to step iv.
 - c. To use **Thales CipherTrust Manager (using KMIP)** as the KMS provider, follow the steps below:
 - i. Enter a unique **Connection Name** for the Key Management service within the project.
 - ii. In the **Address** and **Port** sections, enter the IP of Thales CipherTrust Manager and the port where the KMIP interface is enabled. For example:
 - **Address:** 123.34.3.2
 - **Port:** 5696
 - iii. Upload the **Client Certificate**, **CA certificate**, and **Client Private Key**.
 - iv. If StorageClass encryption is enabled, enter the Unique Identifier to be used for encryption and decryption generated above.
 - v. The **TLS Server** field is optional and used when there is no DNS entry for the KMIP endpoint. For example, **kmip_all_<port>.ciphertrustmanager.local**.
 - d. Select a **Network**.
 - e. Click **Next**.
15. In the **Review and create** page, review the configuration details:
To modify any configuration settings, click **Back**.
 16. Click **Create StorageSystem**.

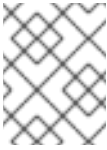
Verification steps

Verifying that the OpenShift Data Foundation cluster is healthy

1. In the OpenShift Web Console, click **Storage** → **Data Foundation**.
2. Click the **Storage Systems** tab and then click on **ocs-storagecluster-storagesystem**.
 - a. In the **Status card** of the **Object** tab, verify that both *Object Service* and *Data Resiliency* have a green tick.
 - b. In the **Details** card, verify that the MCG information is displayed.

Verifying the state of the pods

1. Click **Workloads** → **Pods** from the OpenShift Web Console.
2. Select **openshift-storage** from the **Project** drop-down list and verify that the following pods are in **Running** state.

**NOTE**

If the **Show default projects** option is disabled, use the toggle button to list all the default projects.

Component	Corresponding pods
OpenShift Data Foundation Operator	<ul style="list-style-type: none"> ● ocs-operator-* (1 pod on any storage node) ● ocs-metrics-exporter-* (1 pod on any storage node) ● odf-operator-controller-manager-* (1 pod on any storage node) ● odf-console-* (1 pod on any storage node) ● csi-addons-controller-manager-* (1 pod on any storage node)
Rook-ceph Operator	<p>rook-ceph-operator-*</p> <p>(1 pod on any storage node)</p>
Multicloud Object Gateway	<ul style="list-style-type: none"> ● noobaa-operator-* (1 pod on any storage node) ● noobaa-core-* (1 pod on any storage node) ● noobaa-db-pg-* (1 pod on any storage node) ● noobaa-endpoint-* (1 pod on any storage node) ● noobaa-default-backing-store-noobaa-pod-* (1 pod on any storage node)

CHAPTER 6. VIEW OPENSIFT DATA FOUNDATION TOPOLOGY

The topology shows the mapped visualization of the OpenShift Data Foundation storage cluster at various abstraction levels and also lets you to interact with these layers. The view also shows how the various elements compose the Storage cluster altogether.

Procedure

1. On the OpenShift Web Console, navigate to **Storage → Data Foundation → Topology**.
The view shows the storage cluster and the zones inside it. You can see the nodes depicted by circular entities within the zones, which are indicated by dotted lines. The label of each item or resource contains basic information such as status and health or indication for alerts.
2. Choose a node to view node details on the right-hand panel. You can also access resources or deployments within a node by clicking on the search/preview decorator icon.
3. To view deployment details
 - a. Click the preview decorator on a node. A modal window appears above the node that displays all of the deployments associated with that node along with their statuses.
 - b. Click the **Back to main view** button in the model's upper left corner to close and return to the previous view.
 - c. Select a specific deployment to see more information about it. All relevant data is shown in the side panel.
4. Click the **Resources** tab to view the pods information. This tab provides a deeper understanding of the problems and offers granularity that aids in better troubleshooting.
5. Click the pod links to view the pod information page on OpenShift Container Platform. The link opens in a new window.

CHAPTER 7. UNINSTALLING OPENSIFT DATA FOUNDATION

7.1. Uninstalling OpenShift Data Foundation in Internal-attached devices mode

Use the steps in this section to uninstall OpenShift Data Foundation.

Uninstall Annotations

Annotations on the Storage Cluster are used to change the behavior of the uninstall process. To define the uninstall behavior, the following two annotations have been introduced in the storage cluster:

- **uninstall.ocs.openshift.io/cleanup-policy: delete**
- **uninstall.ocs.openshift.io/mode: graceful**

The following table provides information on the different values that can be used with these annotations:

Table 7.1. uninstall.ocs.openshift.io uninstall annotations descriptions

Annotation	Value	Default	Behavior
cleanup-policy	delete	Yes	Rook cleans up the physical drives and the DataDirHostPath
cleanup-policy	retain	No	Rook does not clean up the physical drives and the DataDirHostPath
mode	graceful	Yes	Rook and NooBaa pauses the uninstall process until the administrator/user removes the Persistent Volume Claims (PVCs) and Object Bucket Claims (OBCs)
mode	forced	No	Rook and NooBaa proceeds with uninstall even if the PVCs/OBCs provisioned using Rook and NooBaa exist respectively

Edit the value of the annotation to change the cleanup policy or the uninstall mode.

```
$ oc -n openshift-storage annotate storagecluster ocs-storagecluster
uninstall.ocs.openshift.io/cleanup-policy="retain" --overwrite
```

```
$ oc -n openshift-storage annotate storagecluster ocs-storagecluster
uninstall.ocs.openshift.io/mode="forced" --overwrite
```


Expected output for both commands:

```
storagecluster.ocs.openshift.io/ocs-storagecluster annotated
```

Prerequisites

- Ensure that the OpenShift Data Foundation cluster is in a healthy state. The uninstall process can fail when some of the pods are not terminated successfully due to insufficient resources or nodes. In case the cluster is in an unhealthy state, contact Red Hat Customer Support before uninstalling OpenShift Data Foundation.
- Ensure that applications are not consuming persistent volume claims (PVCs) or object bucket claims (OBCs) using the storage classes provided by OpenShift Data Foundation.
- If any custom resources (such as custom storage classes, cephblockpools) were created by the admin, they must be deleted by the admin after removing the resources which consumed them.

Procedure

1. Delete the volume snapshots that are using OpenShift Data Foundation.

- a. List the volume snapshots from all the namespaces.

```
$ oc get volumesnapshot --all-namespaces
```

- b. From the output of the previous command, identify and delete the volume snapshots that are using OpenShift Data Foundation.

```
$ oc delete volumesnapshot <VOLUME-SNAPSHOT-NAME> -n <NAMESPACE>
```

<VOLUME-SNAPSHOT-NAME>

Is the name of the volume snapshot

<NAMESPACE>

Is the project namespace

2. Delete PVCs and OBCs that are using OpenShift Data Foundation.

In the default uninstall mode (graceful), the uninstaller waits till all the PVCs and OBCs that use OpenShift Data Foundation are deleted.

If you want to delete the Storage Cluster without deleting the PVCs, you can set the uninstall mode annotation to **forced** and skip this step. Doing so results in orphan PVCs and OBCs in the system.

- a. Delete OpenShift Container Platform monitoring stack PVCs using OpenShift Data Foundation.

See [Removing monitoring stack from OpenShift Data Foundation](#)

- b. Delete OpenShift Container Platform Registry PVCs using OpenShift Data Foundation.

[Removing OpenShift Container Platform registry from OpenShift Data Foundation](#)

- c. Delete OpenShift Container Platform logging PVCs using OpenShift Data Foundation.

[Removing the cluster logging operator from OpenShift Data Foundation](#)

- d. Delete the other PVCs and OBCs provisioned using OpenShift Data Foundation.

- Given below is a sample script to identify the PVCs and OBCs provisioned using OpenShift Data Foundation. The script ignores the PVCs that are used internally by OpenShift Data Foundation.

```
#!/bin/bash

RBD_PROVISIONER="openshift-storage.rbd.csi.ceph.com"
CEPHFS_PROVISIONER="openshift-storage.cephfs.csi.ceph.com"
NOOBAA_PROVISIONER="openshift-storage.noobaa.io/obc"
RGW_PROVISIONER="openshift-storage.ceph.rook.io/bucket"

NOOBAA_DB_PVC="noobaa-db"
NOOBAA_BACKINGSTORE_PVC="noobaa-default-backing-store-noobaa-pvc"

# Find all the OCS StorageClasses
OCS_STORAGECLASSES=$(oc get storageclasses | grep -e
"$RBD_PROVISIONER" -e "$CEPHFS_PROVISIONER" -e
"$NOOBAA_PROVISIONER" -e "$RGW_PROVISIONER" | awk '{print $1}')

# List PVCs in each of the StorageClasses
for SC in $OCS_STORAGECLASSES
do
    echo
    "=====
=="
    echo "$SC StorageClass PVCs and OBCs"
    echo
    "=====
=="
    oc get pvc --all-namespaces --no-headers 2>/dev/null | grep $SC | grep -v -e
"$NOOBAA_DB_PVC" -e "$NOOBAA_BACKINGSTORE_PVC"
    oc get obc --all-namespaces --no-headers 2>/dev/null | grep $SC
    echo
done
```



NOTE

Omit **RGW_PROVISIONER** for cloud platforms.

- Delete the OBCs.

```
$ oc delete obc <obc-name> -n <project-name>
```

<obc-name>

Is the name of the OBC

<project-name>

Is the name of the project

- Delete the PVCs.

```
$ oc delete pvc <pvc-name> -n <project-name>
```

<pvc-name>

Is the name of the PVC

<project-name>

Is the name of the project



NOTE

Ensure that you have removed any custom backing stores, bucket classes, etc., created in the cluster.

3. Delete the Storage System object and wait for the removal of the associated resources.

```
$ oc delete -n openshift-storage storagesystem --all --wait=true
```

4. Check the cleanup pods if the **uninstall.ocs.openshift.io/cleanup-policy** was set to **delete** (default) and ensure that their status is **Completed**.

```
$ oc get pods -n openshift-storage | grep -i cleanup
```

Example output:

NAME	READY	STATUS	RESTARTS	AGE
cluster-cleanup-job-<xx>	0/1	Completed	0	8m35s
cluster-cleanup-job-<yy>	0/1	Completed	0	8m35s
cluster-cleanup-job-<zz>	0/1	Completed	0	8m35s

5. Confirm that the directory **/var/lib/rook** is now empty. This directory is empty only if the **uninstall.ocs.openshift.io/cleanup-policy** annotation was set to **delete** (default).

```
$ for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o jsonpath='{.items[*].metadata.name}'); do oc debug node/${i} -- chroot /host ls -l /var/lib/rook; done
```

6. If encryption was enabled at the time of install, remove **dm-crypt** managed **device-mapper** mapping from the OSD devices on all the OpenShift Data Foundation nodes.

- a. Create a **debug** pod and **chroot** to the host on the storage node.

```
$ oc debug node/<node-name>
```

```
$ chroot /host
```

<node-name>

Is the name of the node

- b. Get Device names and make note of the OpenShift Data Foundation devices.

```
$ dmsetup ls
```

Example output:

```
ocs-deviceset-0-data-0-57snx-block-dmccrypt (253:1)
```

- c. Remove the mapped device.

```
$ cryptsetup luksClose --debug --verbose ocs-deviceset-0-data-0-57snx-block-dmccrypt
```



IMPORTANT

If the above command gets stuck due to insufficient privileges, run the following commands:

- Press **CTRL+Z** to exit the above command.
- Find PID of the process which was stuck.

```
$ ps -ef | grep crypt
```

- Terminate the process using **kill** command.

```
$ kill -9 <PID>
```

<PID>

Is the process ID

- Verify that the device name is removed.

```
$ dmsetup ls
```

7. Delete the namespace and wait till the deletion is complete. You need to switch to another project if **openshift-storage** is the active project.

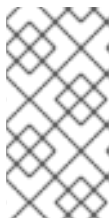
For example:

```
$ oc project default
```

```
$ oc delete project openshift-storage --wait=true --timeout=5m
```

The project is deleted if the following command returns a NotFound error.

```
$ oc get project openshift-storage
```



NOTE

While uninstalling OpenShift Data Foundation, if **namespace** is not deleted completely and remains in **Terminating** state, perform the steps in [Troubleshooting and deleting remaining resources during Uninstall](#) to identify objects that are blocking the namespace from being terminated.

8. Delete local storage operator configurations if you have deployed OpenShift Data Foundation using local storage devices. See [Removing local storage operator configurations](#).
9. Unlabel the storage nodes.

```
$ oc label nodes --all cluster.ocs.openshift.io/openshift-storage-
```

```
$ oc label nodes --all topology.rook.io/rack-
```

- Remove the OpenShift Data Foundation taint if the nodes were tainted.

```
$ oc adm taint nodes --all node.ocs.openshift.io/storage-
```

- Confirm all the Persistent volumes (PVs) provisioned using OpenShift Data Foundation are deleted. If there is any PV left in the **Released** state, delete it.

```
$ oc get pv
```

```
$ oc delete pv <pv-name>
```

<pv-name>

Is the name of the PV

- Remove the **CustomResourceDefinitions**.

```
$ oc delete crd backingstores.noobaa.io bucketclasses.noobaa.io
cephblockpools.ceph.rook.io cephclusters.ceph.rook.io cephfilesystems.ceph.rook.io
cephnfses.ceph.rook.io cephobjectstores.ceph.rook.io cephobjectstoreusers.ceph.rook.io
noobaas.noobaa.io ocsinitializations.ocs.openshift.io storageclusters.ocs.openshift.io
cephclients.ceph.rook.io cephobjectrealms.ceph.rook.io cephobjectzonegroups.ceph.rook.io
cephobjectzones.ceph.rook.io cephrbdmirrors.ceph.rook.io storagesystems.odf.openshift.io --
wait=true --timeout=5m
```

- To ensure that OpenShift Data Foundation is uninstalled completely, on the OpenShift Container Platform Web Console,
 - Click **Storage**.
 - Verify that **OpenShift Data Foundation** no longer appears under Storage.

7.1.1. Removing local storage operator configurations

Use the instructions in this section only if you have deployed OpenShift Data Foundation using local storage devices.



NOTE

For OpenShift Data Foundation deployments only using **localvolume** resources, go directly to step 8.

Procedure

- Identify the **LocalVolumeSet** and the corresponding **StorageClassName** being used by OpenShift Data Foundation.

```
$ oc get localvolumesets.local.storage.openshift.io -n openshift-local-storage
```

- Set the variable SC to the **StorageClass** providing the **LocalVolumeSet**.

```
$ export SC="<StorageClassName>"
```

- List and note the devices to be cleaned up later. In order to list the device ids of the disks, follow the procedure mentioned here, See [Find the available storage devices](#).

Example output:

```
/dev/disk/by-id/scsi-360050763808104bc28000000000000eb
/dev/disk/by-id/scsi-360050763808104bc28000000000000ef
/dev/disk/by-id/scsi-360050763808104bc28000000000000f3
```

- Delete the **LocalVolumeSet**.

```
$ oc delete localvolumesets.local.storage.openshift.io <name-of-volumeset> -n openshift-
local-storage
```

- Delete the local storage PVs for the given **StorageClassName**.

```
$ oc get pv | grep $SC | awk '{print $1}' | xargs oc delete pv
```

- Delete the **StorageClassName**.

```
$ oc delete sc $SC
```

- Delete the symlinks created by the **LocalVolumeSet**.

```
[[ ! -z $SC ]] && for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o
jsonpath='{.items[*].metadata.name }'); do oc debug node/${i} -- chroot /host rm -rfv
/mnt/local-storage/${SC}/; done
```

- Delete **LocalVolumeDiscovery**.

```
$ oc delete localvolumediscovery.local.storage.openshift.io/auto-discover-devices -n
openshift-local-storage
```

- Remove the **LocalVolume** resources (if any).

Use the following steps to remove the **LocalVolume** resources that were used to provision PVs in the current or previous OpenShift Data Foundation version. Also, ensure that these resources are not being used by other tenants on the cluster.

For each of the local volumes, do the following:

- Identify the **LocalVolume** and the corresponding **StorageClassName** being used by OpenShift Data Foundation.

```
$ oc get localvolume.local.storage.openshift.io -n openshift-local-storage
```

- Set the variable LV to the name of the LocalVolume and variable SC to the name of the StorageClass

For example:

```
$ LV=local-block
$ SC=localblock
```

- c. List and note the devices to be cleaned up later.

```
$ oc get localvolume -n openshift-local-storage $LV -o jsonpath='{
.spec.storageClassDevices[].devicePaths[] }{"\n"}
```

Example output:

```
/dev/sdb
/dev/sdc
/dev/sdd
/dev/sde
```

- d. Delete the local volume resource.

```
$ oc delete localvolume -n openshift-local-storage --wait=true $LV
```

- e. Delete the remaining PVs and StorageClasses if they exist.

```
$ oc delete pv -l storage.openshift.com/local-volume-owner-name=${LV} --wait --
timeout=5m
$ oc delete storageclass $SC --wait --timeout=5m
```

- f. Clean up the artifacts from the storage nodes for that resource.

```
$ [[ ! -z $SC ]] && for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o
jsonpath='{ .items[*].metadata.name }'); do oc debug node/${i} -- chroot /host rm -rfv
/mnt/local-storage/${SC}/; done
```

Example output:

```
Starting pod/node-xxx-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
Starting pod/node-yyy-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
Starting pod/node-zzz-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
```

10. Wipe the disks for each of the local volumesets or local volumes listed in step 1 and 8 respectively so that they can be reused.

- a. List the storage nodes.

```
oc get nodes -l cluster.ocs.openshift.io/openshift-storage=
```

Example output:

```
NAME      STATUS  ROLES  AGE   VERSION
node-xxx  Ready   worker  4h45m v1.18.3+6c42de8
node-yyy  Ready   worker  4h46m v1.18.3+6c42de8
node-zzz  Ready   worker  4h45m v1.18.3+6c42de8
```

- b. Obtain the node console and execute **chroot /host** command when the prompt appears.

```
$ oc debug node/node-xxx
Starting pod/node-xxx-debug ...
To use host binaries, run `chroot /host`
Pod IP: w.x.y.z
If you don't see a command prompt, try pressing enter.
sh-4.2# chroot /host
```

- c. Store the disk paths in the **DISKS** variable within quotes. For the list of disk paths, see step 3 and step 8.c for local volumeset and local volume respectively.

Example output:

```
sh-4.4# DISKS="/dev/disk/by-id/scsi-360050763808104bc28000000000000eb
/dev/disk/by-id/scsi-360050763808104bc28000000000000ef /dev/disk/by-id/scsi-
360050763808104bc28000000000000f3 "
or
sh-4.2# DISKS="/dev/sdb /dev/sdc /dev/sdd /dev/sde ".
```

- d. Run **sgdisk --zap-all** on all the disks.

```
sh-4.4# for disk in $DISKS; do sgdisk --zap-all $disk;done
```

Example output:

```
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
```

- e. Exit the shell and repeat for the other nodes.

```
sh-4.4# exit
exit
sh-4.2# exit
```



```
exit
Removing debug pod ...
```

11. Delete the **openshift-local-storage** namespace and wait till the deletion is complete. You will need to switch to another project if the **openshift-local-storage** namespace is the active project.

For example:

```
$ oc project default
$ oc delete project openshift-local-storage --wait=true --timeout=5m
```

The project is deleted if the following command returns a **NotFound** error.

```
$ oc get project openshift-local-storage
```

7.2. Removing monitoring stack from OpenShift Data Foundation

Use this section to clean up the monitoring stack from OpenShift Data Foundation.

The Persistent Volume Claims (PVCs) that are created as a part of configuring the monitoring stack are in the **openshift-monitoring** namespace.

Prerequisites

- PVCs are configured to use the OpenShift Container Platform monitoring stack. For more information, see [configuring monitoring stack](#).

Procedure

1. List the pods and PVCs that are currently running in the **openshift-monitoring** namespace.

```
$ oc get pod,pvc -n openshift-monitoring
```

Example output:

NAME	READY	STATUS	RESTARTS	AGE
pod/alertmanager-main-0	3/3	Running	0	8d
pod/alertmanager-main-1	3/3	Running	0	8d
pod/alertmanager-main-2	3/3	Running	0	8d
pod/cluster-monitoring-operator-84457656d-pkrxm	1/1	Running	0	8d
pod/grafana-79ccf6689f-2ll28	2/2	Running	0	8d
pod/kube-state-metrics-7d86fb966-rvd9w	3/3	Running	0	8d
pod/node-exporter-25894	2/2	Running	0	8d
pod/node-exporter-4dsd7	2/2	Running	0	8d
pod/node-exporter-6p4zc	2/2	Running	0	8d
pod/node-exporter-jbjvg	2/2	Running	0	8d
pod/node-exporter-jj4t5	2/2	Running	0	6d18h
pod/node-exporter-k856s	2/2	Running	0	6d18h
pod/node-exporter-rf8gn	2/2	Running	0	8d
pod/node-exporter-rmb5m	2/2	Running	0	6d18h
pod/node-exporter-zj7kx	2/2	Running	0	8d

```

pod/openshift-state-metrics-
59dbd4f654-4clng      3/3   Running 0      8d
pod/prometheus-adapter-
5df5865596-k8dzn     1/1   Running 0      7d23h
pod/prometheus-adapter-
5df5865596-n2gj9     1/1   Running 0      7d23h
pod/prometheus-k8s-0   6/6   Running 1      8d
pod/prometheus-k8s-1   6/6   Running 1      8d
pod/prometheus-operator-
55cfb858c9-c4zd9     1/1   Running 0      6d21h
pod/telemeter-client-
78fc8fc97d-2rgfp     3/3   Running 0      8d

```

```

NAME                                     STATUS VOLUME
CAPACITY ACCESS MODES STORAGECLASS          AGE
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-0 Bound pvc-0d519c4f-
15a5-11ea-baa0-026d231574aa 40Gi RWO          ocs-storagecluster-ceph-rbd 8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-1 Bound pvc-
0d5a9825-15a5-11ea-baa0-026d231574aa 40Gi RWO          ocs-storagecluster-ceph-
rbd 8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-2 Bound pvc-
0d6413dc-15a5-11ea-baa0-026d231574aa 40Gi RWO          ocs-storagecluster-ceph-
rbd 8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-0 Bound pvc-0b7c19b0-
15a5-11ea-baa0-026d231574aa 40Gi RWO          ocs-storagecluster-ceph-rbd 8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-1 Bound pvc-0b8aed3f-
15a5-11ea-baa0-026d231574aa 40Gi RWO          ocs-storagecluster-ceph-rbd 8d

```

2. Edit the monitoring **configmap**.

```
$ oc -n openshift-monitoring edit configmap cluster-monitoring-config
```

Remove any **config** sections that reference the OpenShift Data Foundation storage classes as shown in the following example and save it.

Before editing

```
.  
.   
.   
apiVersion: v1  
data:  
  config.yaml: |  
    alertmanagerMain:  
      volumeClaimTemplate:  
        metadata:  
          name: my-alertmanager-claim  
        spec:  
          resources:  
            requests:  
              storage: 40Gi  
          storageClassName: ocs-storagecluster-ceph-rbd  
  prometheusK8s:  
    volumeClaimTemplate:  
      metadata:  
        name: my-prometheus-claim  
      spec:  
        resources:  
          requests:  
            storage: 40Gi  
        storageClassName: ocs-storagecluster-ceph-rbd  
kind: ConfigMap  
metadata:  
  creationTimestamp: "2019-12-02T07:47:29Z"  
  name: cluster-monitoring-config  
  namespace: openshift-monitoring  
  resourceVersion: "22110"  
  selfLink: /api/v1/namespaces/openshift-monitoring/configmaps/cluster-monitoring-config  
  uid: fd6d988b-14d7-11ea-84ff-066035b9efa8  
.   
.   
.   

```

After editing

```

.
.
.
apiVersion: v1
data:
  config.yaml: |
kind: ConfigMap
metadata:
  creationTimestamp: "2019-11-21T13:07:05Z"
  name: cluster-monitoring-config
  namespace: openshift-monitoring
  resourceVersion: "404352"
  selfLink: /api/v1/namespaces/openshift-monitoring/configmaps/cluster-monitoring-config
  uid: d12c796a-0c5f-11ea-9832-063cd735b81c
.
.
.

```

In this example, **alertmanagerMain** and **prometheusK8s** monitoring components are using the OpenShift Data Foundation PVCs.

3. Delete the relevant PVCs. Make sure you delete all the PVCs that are consuming the storage classes.

```
$ oc delete -n openshift-monitoring pvc <pvc-name> --wait=true --timeout=5m
```

<pvc-name>

Is the name of the PVC

7.3. Removing OpenShift Container Platform registry from OpenShift Data Foundation

Use this section to clean up the OpenShift Container Platform registry from OpenShift Data Foundation. If you want to configure an alternative storage, see [Image registry](#).

The Persistent Volume Claims (PVCs) that are created as a part of configuring the OpenShift Container Platform registry are in the **openshift-image-registry** namespace.

Prerequisites

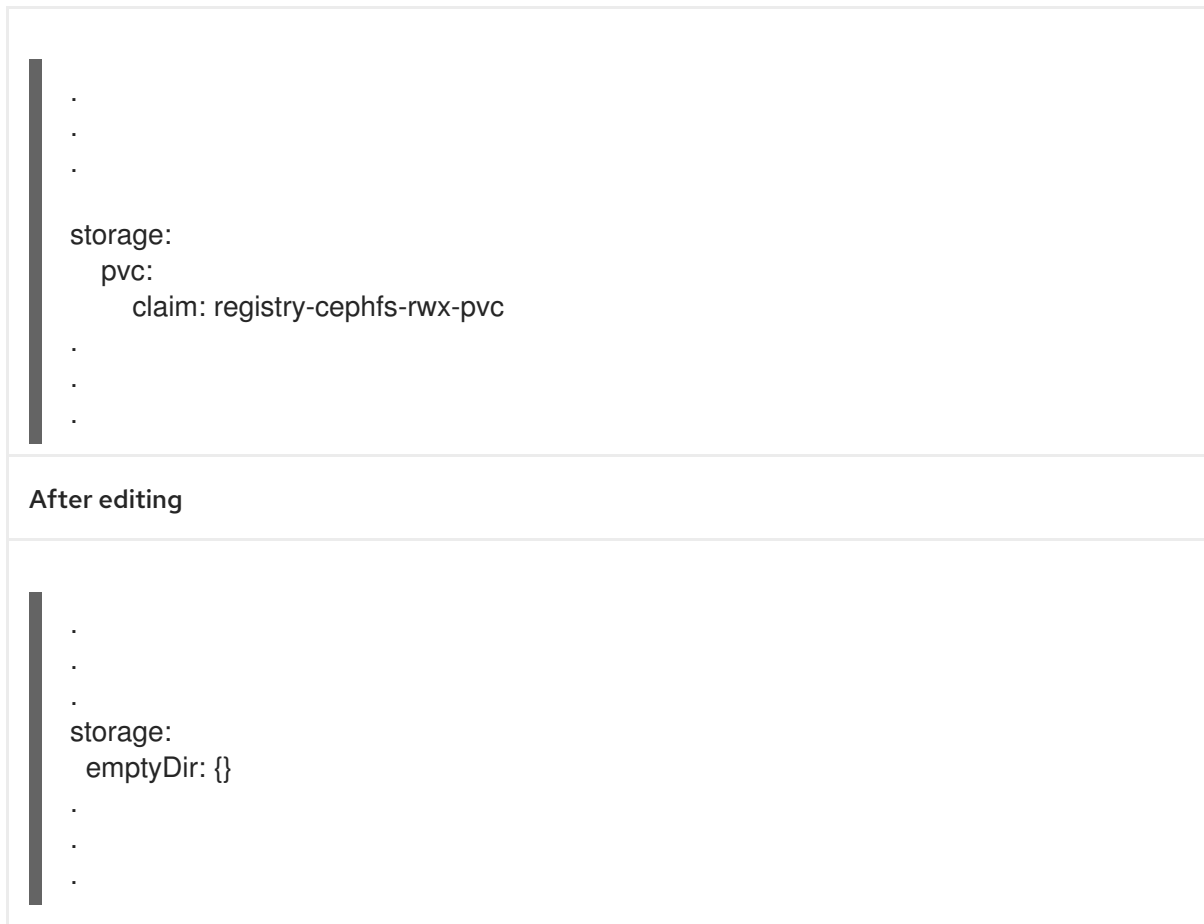
- The image registry must have been configured to use an OpenShift Data Foundation PVC.

Procedure

1. Edit the **configs.imageregistry.operator.openshift.io** object and remove the content in the **storage** section.

```
$ oc edit configs.imageregistry.operator.openshift.io
```

Before editing



In this example, the PVC is called **registry-cephfs-rwx-pvc**, which is now safe to delete.

2. Delete the PVC.

```
$ oc delete pvc <pvc-name> -n openshift-image-registry --wait=true --timeout=5m
```

<pvc-name>

Is the name of the PVC

7.4. Removing the cluster logging operator from OpenShift Data Foundation

Use this section to clean up the cluster logging operator from OpenShift Data Foundation.

The Persistent Volume Claims (PVCs) that are created as a part of configuring the cluster logging operator are in the **openshift-logging** namespace.

Prerequisites

- The cluster logging instance should have been configured to use the OpenShift Data Foundation PVCs.

Procedure

1. Remove the **ClusterLogging** instance in the namespace.

```
$ oc delete clusterlogging instance -n openshift-logging --wait=true --timeout=5m
```

The PVCs in the **openshift-logging** namespace are now safe to delete.

2. Delete the PVCs.

```
┆ $ oc delete pvc <pvc-name> -n openshift-logging --wait=true --timeout=5m
```

<pvc-name>

Is the name of the PVC