Red Hat OpenShift Container Storage 4.3

Deploying OpenShift Container Storage

How to install and set up your environment

Last Updated: 2020-12-08

Red Hat OpenShift Container Storage 4.3 Deploying OpenShift Container Storage

How to install and set up your environment

Legal Notice

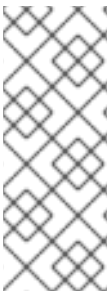
Abstract

Read this document for instructions on installing Red Hat OpenShift Container Storage 4.3.

Table of Contents

CHAPTER 1. DEPLOYING OPENSHIFT CONTAINER STORAGE

OpenShift Container Storage 4.3, installation is supported only on existing Red Hat OpenShift Container Platform (OCP) worker nodes. Follow the instructions in Section 1.1, "Deploying Red Hat OpenShift Container Storage on an existing Red Hat OpenShift Container Platform" to deploy OpenShift Container Storage.



NOTE

When you install OpenShift Container Storage in a restricted network environment, you need to apply a custom Network Time Protocol (NTP) configuration to the nodes, because by default, internet connectivity is assumed in OpenShift Container Platform and **chronyd** is configured to use ***.rhel.pool.ntp.org** servers. See https://access.redhat.com/solutions/4828941 and Configuring chrony time service for more details.

1.1. DEPLOYING RED HAT OPENSHIFT CONTAINER STORAGE ON AN EXISTING RED HAT OPENSHIFT CONTAINER PLATFORM

The deployment process consists of two main parts:

1. Install the OpenShift Container Storage Operator by following the instructions in Section 1.1.1, "Installing Red Hat OpenShift Container Storage Operator using the Operator Hub".
2. Create the OpenShift Container Storage service by following the instructions in Section 1.1.2, "Creating an OpenShift Container Storage service".

For Red Hat Enterprise Linux based hosts in a user provisioned infrastructure (UPI), you need to enable container access to the underlying file system by following the instructions in Section 1.1.3, "Enabling file system access for containers on Red Hat Enterprise Linux based nodes".

1.1.1. Installing Red Hat OpenShift Container Storage Operator using the Operator Hub

You can install Red Hat OpenShift Container Storage using the Red Hat OpenShift Container Platform Operator Hub on Amazon Web Services (AWS) and VMware vSphere platforms. For information about the hardware and software requirements, see Planning your deployment.

Prerequisites

- You must be logged into OpenShift Container Platform (OCP) cluster.
- You must have at least three worker nodes in OCP cluster.
- You must create a namespace called **openshift-storage** as follows:
    1. Click **Administration → Namespaces** in the left pane of the OpenShift Web Console.
    2. Click **Create Namespace**.
    3. In the Create Namespace dialog box, enter **openshift-storage** for Name and **openshift.io/cluster-monitoring=true** for Labels. This label is required to get the dashboards.
    4. Select **No restrictions** option for **Default Network Policy**.

5. Click **Create**.

NOTE

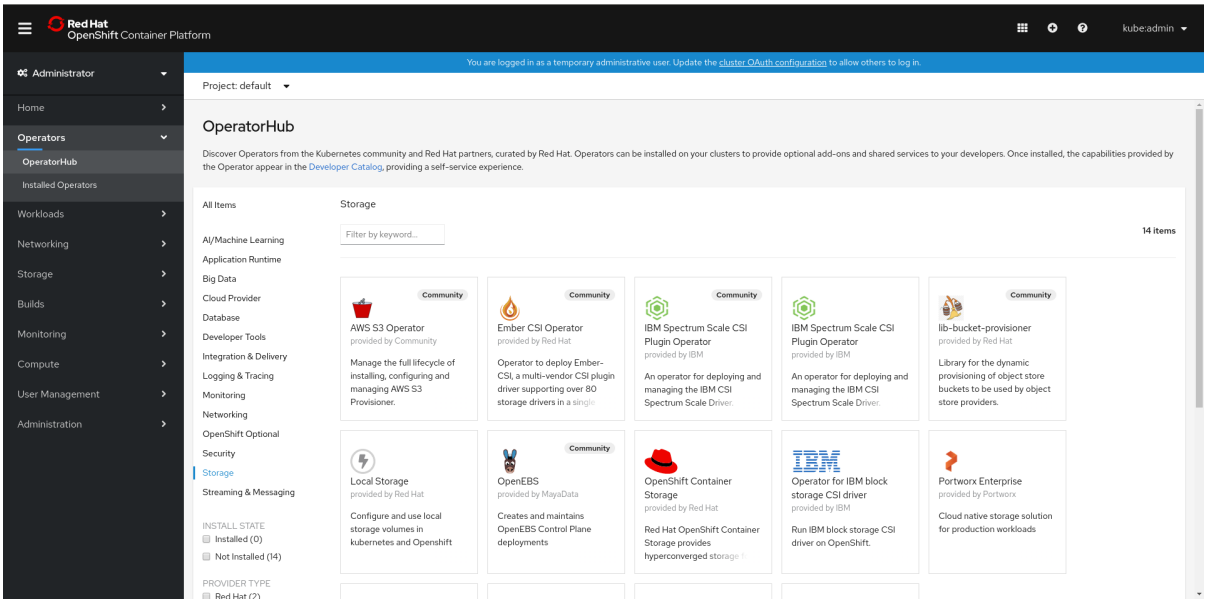
When you need to override the cluster-wide default node selector for OpenShift Container Storage, you can use the following command in command line interface to specify a blank node selector for the **openshift-storage** namespace:

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

Procedure

1. Click **Operators → OperatorHub** in the left pane of the OpenShift Web Console.

Figure 1.1. List of operators in the Operator Hub

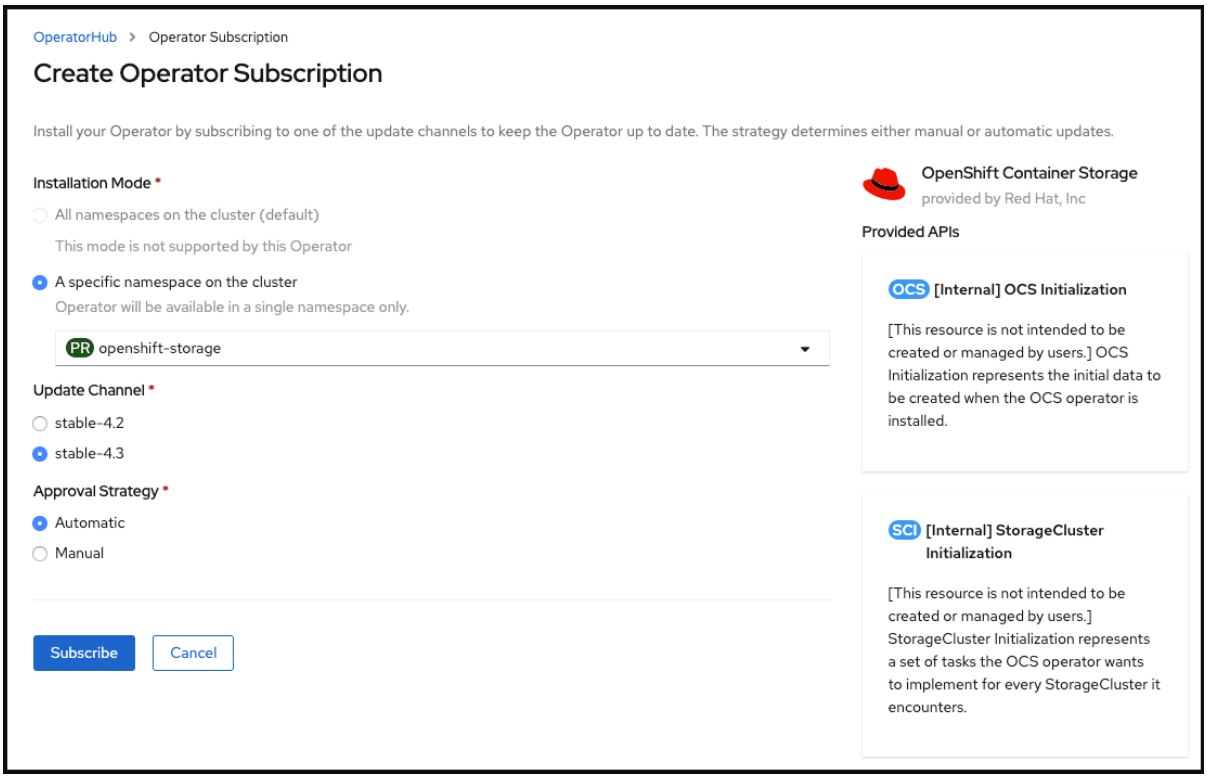


2. Search for **OpenShift Container Storage Operator** from the list of operators and click on it.

3. On the OpenShift Container Storage Operator page, click **Install**.

4. On the Create Operator Subscription page, the Installation Mode, Update Channel, and Approval Strategy options can be set.

Figure 1.2. Create Operator Subscription page



a. Select **A specific namespace on the cluster**for the Installation Mode option.

   - Select **openshift-storage** namespace from the drop down menu.

b. Select **stable-4.3** as the update channel based on your requirement.

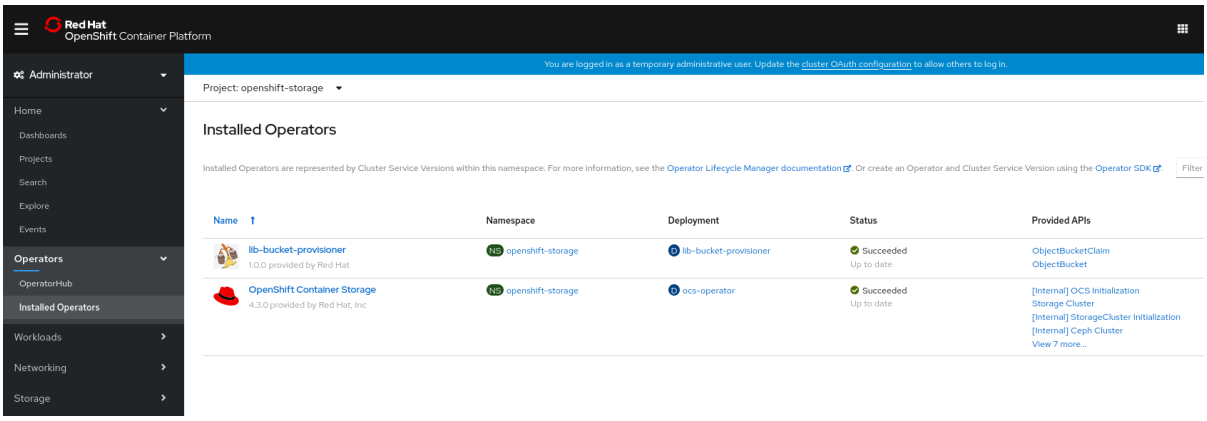
c. Select an Approval Strategy:

   - **Automatic** specifies that you want OpenShift Container Platform to upgrade OpenShift Container Storage automatically.

   - **Manual** specifies that you want to have control to upgrade OpenShift Container Storage manually.

5. Click **Subscribe**.

Figure 1.3. Installed operators



The Installed Operators page is displayed with the status of the operator.

Verification steps

- Verify that the **lib-bucket-provisioner** and **OpenShift Container Storage Operator** show the Status as **Succeeded**.

1.1.2. Creating an OpenShift Container Storage service

You need to create a new OpenShift Container Storage service after you install OpenShift Container Storage operator on a user provisioned cloud for both Amazon Web Services (AWS) and VMware vSphere platforms.

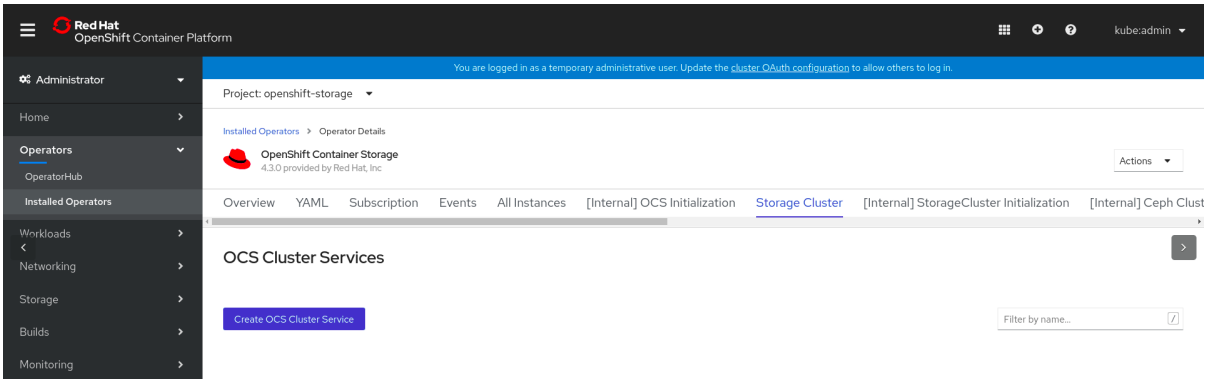
Prerequisites

- OpenShift Container Storage operator must be installed from the Operator Hub. For more information, see Installing OpenShift Container Storage Operator using the Operator Hub .

Procedure

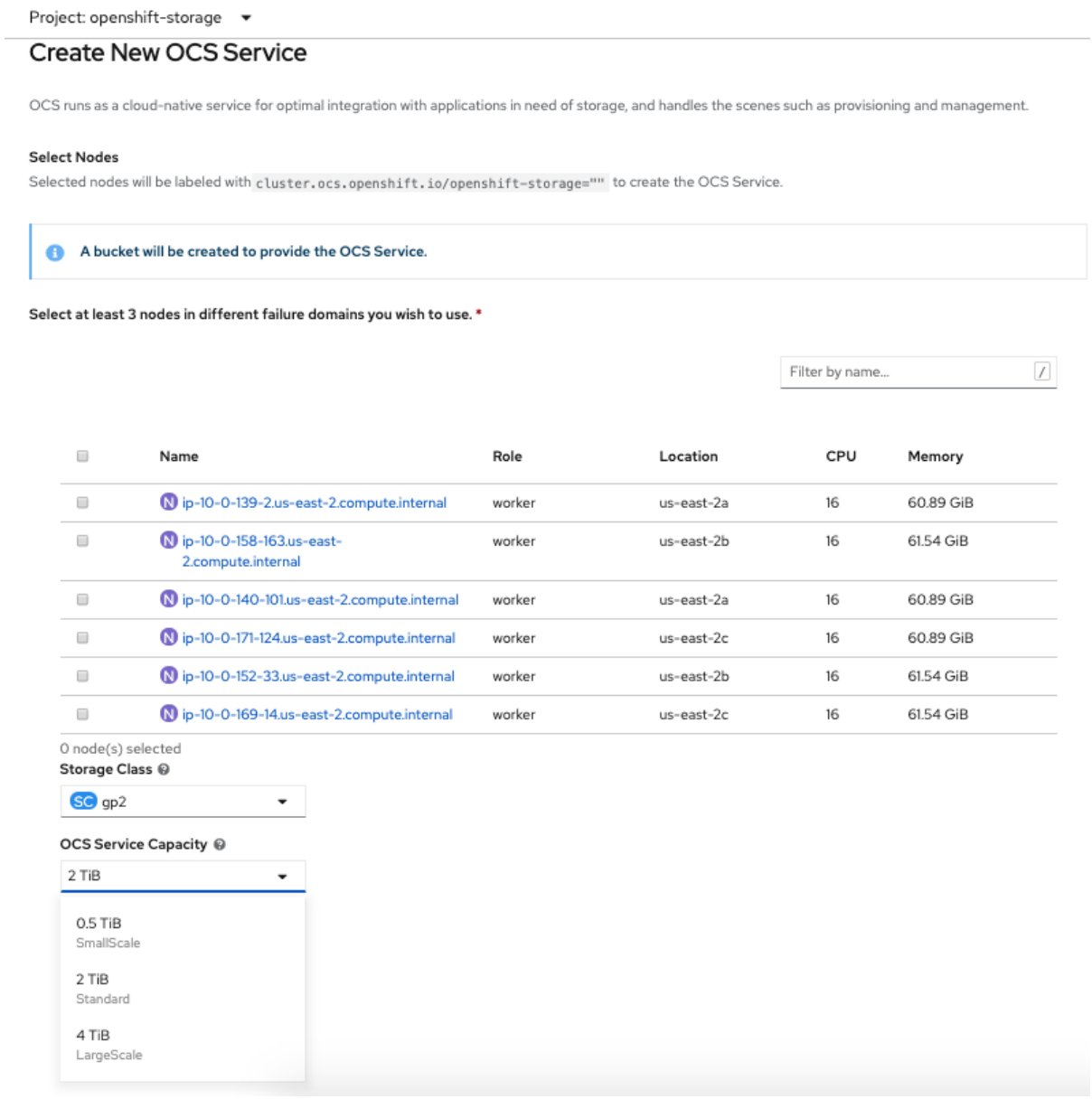
1. Click **Operators → Installed Operators** from the left pane of the OpenShift Web Console to view the installed operators.
2. On the Installed Operator page, select **openshift-storage** from the **Project** drop down list to switch to the **openshift-storage** project.
3. Click **OpenShift Container Storage Operator**.
   OpenShift Container Storage operator creates a *OCSInitialization* resource automatically.
4. On the OpenShift Container Storage Operator page, scroll right and click the **Storage Cluster** tab.

   Figure 1.4. OpenShift Container Storage Operator page



5. On the **OCS Cluster Services** page, click **Create OCS Cluster Service**.

Figure 1.5. Create New OCS Service page

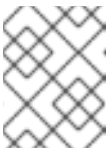


6. On the **Create New OCS Service** page, perform the following:

   a. Select at least three worker nodes from the available list of nodes for the use of OpenShift Container Storage service. Ensure that the nodes are in different **Location**.

   b. **Storage Class** is set by default to **gp2** for AWS and **thin** for VMware.

   c. Select **OCS Service Capacity** from drop down list.



NOTE

Once you select the initial storage capacity here, you can add more capacity only in this increment.

7. Click **Create**.
   The **Create** button is enabled only after you select three nodes. A new storage cluster of three volumes will be created with one volume per worker node. The default configuration uses a replication factor of 3.

Verification steps

- To verify that OpenShift Container Storage is successfully installed, see Verifying your OpenShift Container Storage installation.

1.1.3. Enabling file system access for containers on Red Hat Enterprise Linux based nodes

Deploying OpenShift Container Platform on a Red Hat Enterprise Linux base in a user provisioned infrastructure (UPI) does not automatically provide container access to the underlying Ceph file system. This is a bug tracked by RHSTOR-787.

NOTE

This process is not necessary for hosts based on Red Hat Enterprise Linux CoreOS.

Procedure

Perform the following steps on each node in your cluster.

1. Log in to the Red Hat Enterprise Linux based node and open a terminal.
2. Verify that the node has access to the rhel-7-server-extras-rpms repository.

```
# subscription-manager repos --list-enabled | grep rhel-7-server
```

   If you do not see both **rhel-7-server-rpms** and **rhel-7-server-extras-rpms** in the output, or if there is no output, run the following commands to enable each repository.

```
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-7-server-extras-rpms
```

3. Install the required packages.

```
# yum install -y policycoreutils container-selinux
```

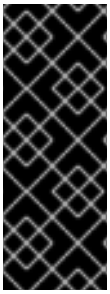
4. Persistently enable container use of the Ceph file system in SELinux.

```
# setsebool -P container_use_cephfs on
```

5. Verify that containers can now access OpenShift Container Storage hosted on this node.

1.2. INSTALLING OPENSHIFT CONTAINER STORAGE USING LOCAL STORAGE DEVICES

Use this section to install OpenShift Container Storage on bare metal, Amazon EC2, and VMware infrastructures where OpenShift Container Platform is already installed.



IMPORTANT

Installing OpenShift Container Storage on bare metal, Amazon EC2, and VMware using local storage operator is a Technology Preview feature. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

To install OpenShift Container Storage using local storage devices, perform the following steps:

1. Understand the requirements for installing OpenShift Container Storage using local storage devices.
2. Install Red Hat OpenShift Container Storage Operator .
3. Install Local Storage Operator .
4. Find the available storage devices .
5. Create OpenShift Container Storage cluster based on your requirement:
   - For Amazon EC2, follow the instructions in Creating OpenShift Container Storage cluster on Amazon EC2.
   - For VMware, follow the instructions in Creating OpenShift Container Storage cluster on VMware.
   - For bare metal, follow the instructions in Creating OpenShift Container Storage cluster on bare metal.

1.2.1. Requirements for installing OpenShift Container Storage using local storage devices

- You must have at least three OpenShift Container Platform worker nodes in the cluster with locally attached storage devices on each of them.
  - Each of the three worker nodes must have at least one raw block device available to be used by OpenShift Container Storage.
  - For minimum starting node requirements, see Node Requirements section in Planning guide.
- The devices to be used must be empty, that is, there should be no PVs, VGs, or LVs remaining on the disks.
- You must have a minimum of three labeled nodes.
  - Each worker node that has local storage devices to be used by OpenShift Container Storage must have a specific label to deploy OpenShift Container Storage pods. To label the nodes, use the following command:

    ```
    $ oc label nodes <NodeName> cluster.ocs.openshift.io/openshift-storage=''
    ```

- There should not be any storage providers managing locally mounted storage on the storage nodes that would conflict with the use of Local Storage Operator for Red Hat OpenShift Container Storage.

1.2.2. Installing Red Hat OpenShift Container Storage Operator using the Operator Hub

You can install Red Hat OpenShift Container Storage using the Red Hat OpenShift Container Platform Operator Hub on Amazon Web Services (AWS) and VMware vSphere platforms. For information about the hardware and software requirements, see Planning your deployment.

Prerequisites

- You must be logged into OpenShift Container Platform (OCP) cluster.
- You must have at least three worker nodes in OCP cluster.
- You must create a namespace called **openshift-storage** as follows:
    1. Click **Administration → Namespaces** in the left pane of the OpenShift Web Console.
    2. Click **Create Namespace**.
    3. In the Create Namespace dialog box, enter **openshift-storage** for Name and **openshift.io/cluster-monitoring=true** for Labels. This label is required to get the dashboards.
    4. Select **No restrictions** option for **Default Network Policy**.
    5. Click **Create**.

NOTE

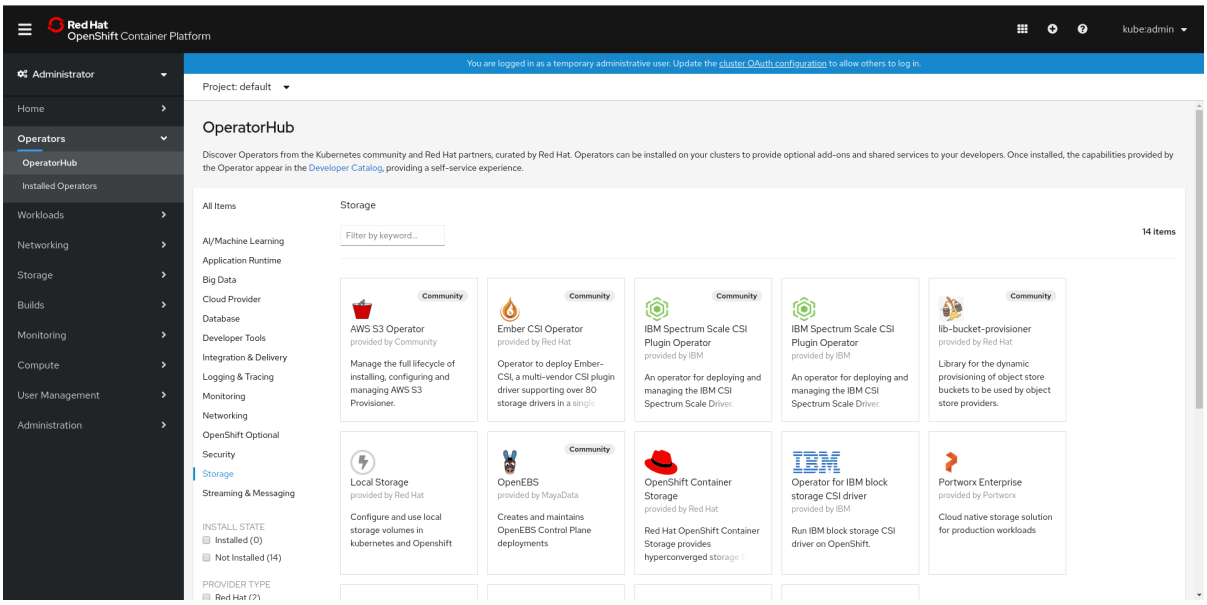
When you need to override the cluster-wide default node selector for OpenShift Container Storage, you can use the following command in command line interface to specify a blank node selector for the **openshift-storage** namespace:

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

Procedure

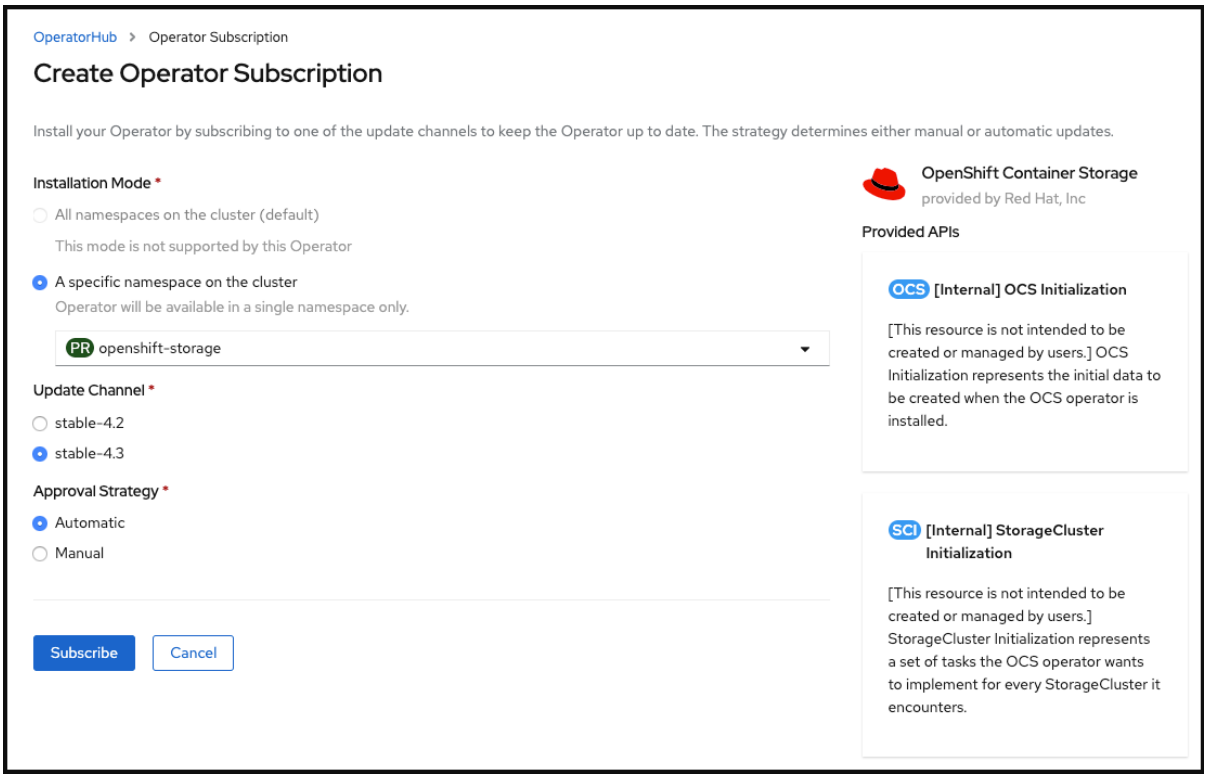
1. Click **Operators → OperatorHub** in the left pane of the OpenShift Web Console.

Figure 1.6. List of operators in the Operator Hub



2. Search for **OpenShift Container Storage Operator** from the list of operators and click on it.
3. On the OpenShift Container Storage Operator page, click **Install**.
4. On the Create Operator Subscription page, the Installation Mode, Update Channel, and Approval Strategy options can be set.

Figure 1.7. Create Operator Subscription page



a. Select **A specific namespace on the cluster** for the Installation Mode option.

   - Select **openshift-storage** namespace from the drop down menu.

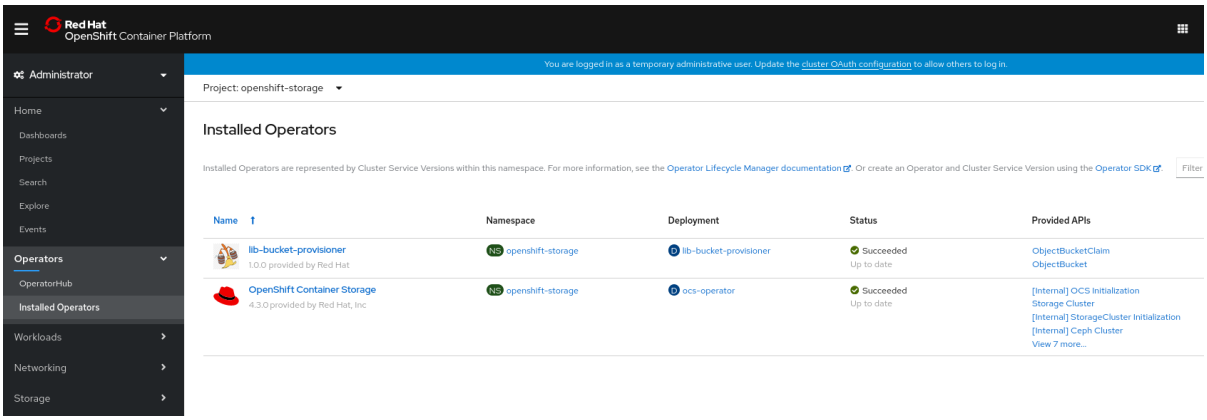
b. Select **stable-4.3** as the update channel based on your requirement.

c. Select an Approval Strategy:

- **Automatic** specifies that you want OpenShift Container Platform to upgrade OpenShift Container Storage automatically.
- **Manual** specifies that you want to have control to upgrade OpenShift Container Storage manually.

5. Click **Subscribe**.

Figure 1.8. Installed operators



The Installed Operators page is displayed with the status of the operator.

Verification steps

- Verify that the **lib-bucket-provisioner** and **OpenShift Container Storage Operator** show the Status as **Succeeded**.

1.2.3. Installing Local Storage Operator

Use this procedure to install Local Storage Operator from the Operator Hub before creating OpenShift Container Storage clusters on local storage devices in Amazon EC2, VMware, and bare metal infrastructures.

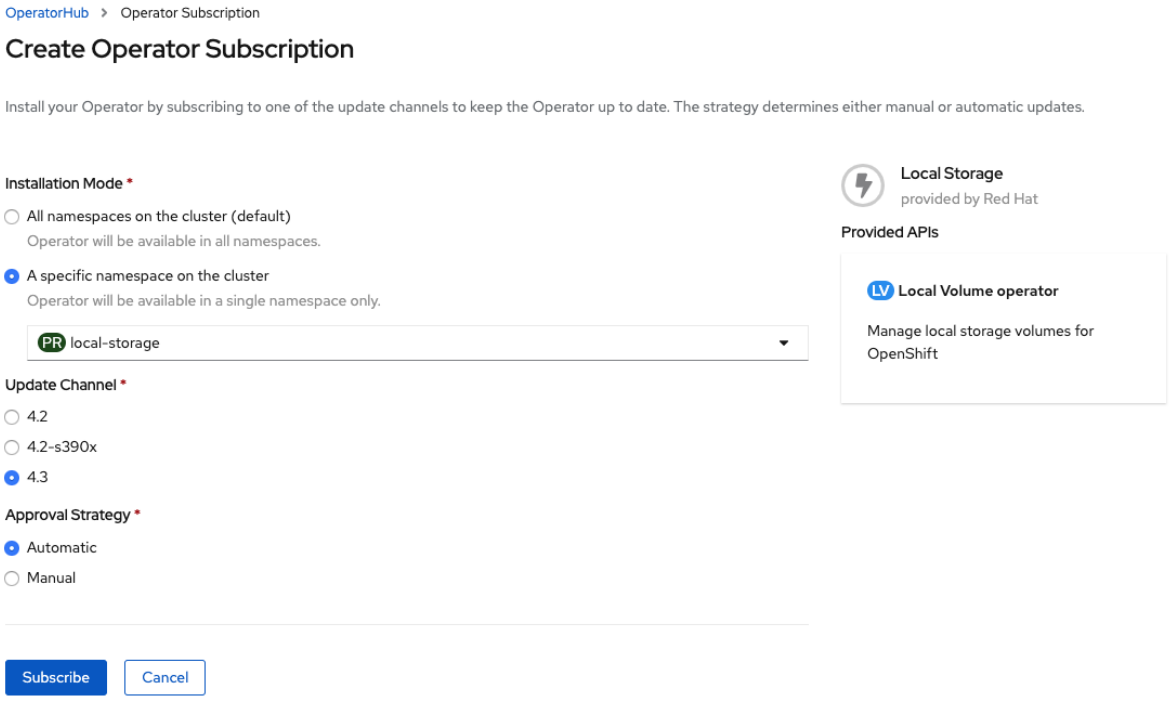
Prerequisites

- Create a namespace called **local-storage** as follows:
  a. Click **Administration → Namespaces** in the left pane of the OpenShift Web Console.
  b. Click **Create Namespace**.
  c. In the Create Namespace dialog box, enter **local-storage** for Name.
  d. Select **No restrictions** option for **Default Network Policy**.
  e. Click **Create**.

Procedure

1. Click **Operators → OperatorHub** in the left pane of the OpenShift Web Console.
2. Search for **Local Storage Operator** from the list of operators and click on it.
3. Click **Install**.

Figure 1.9. Create Operator Subscription page



4. Select **A specific namespace on the cluster**for the Installation Mode option.
   - Select **local-storage** namespace from the drop down menu.
5. Select a desired value for the **Update Channel** option.
6. Select the desired **Approval Strategy**.
7. Click **Subscribe**.
8. Verify that the Local Storage Operator show the Status as **Succeeded**.

1.2.4. Finding available storage devices

Use this procedure to identify the device name for each of the three or more worker nodes that you have labeled with OpenShift Container Storage label, **cluster.ocs.openshift.io/openshift-storage=''** before creating PVs for bare metal, Amazon EC2, or VMware storage devices.

Procedure

1. List and verify the name of the worker nodes with the OpenShift Container Storage label.

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

Example output:

```
NAME                           STATUS   ROLES    AGE     VERSION
ip-10-0-135-71.us-east-2.compute.internal    Ready    worker   6h45m   v1.16.2
ip-10-0-145-125.us-east-2.compute.internal   Ready    worker   6h45m   v1.16.2
ip-10-0-160-91.us-east-2.compute.internal    Ready    worker   6h45m   v1.16.2
```

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

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

Example output:

```
$ oc debug node/ip-10-0-135-71.us-east-2.compute.internal
Starting pod/ip-10-0-135-71us-east-2computeinternal-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
xvda                         202:0    0   120G  0 disk
|-xvda1                      202:1    0   384M  0 part /boot
|-xvda2                      202:2    0   127M  0 part /boot/efi
|-xvda3                      202:3    0     1M  0 part
`-xvda4                      202:4    0 119.5G  0 part
  `-coreos-luks-root-nocrypt 253:0    0 119.5G  0 dm   /sysroot
nvme0n1                      259:0    0   1.7T  0 disk
nvme1n1                      259:1    0   1.7T  0 disk
```

In this example, the local devices available are **nvme0n1** and **nvme1n1**.

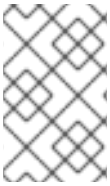
3. Find the unique **by-id** device name depending on the hardware serial number for each device.

```
sh-4.4# ls -l /dev/disk/by-id/
total 0
lrwxrwxrwx. 1 root root 10 Mar 17 16:24 dm-name-coreos-luks-root-nocrypt -> ../../dm-0
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC -> ../../nvme0n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS60382E5D7441494EC -> ../../nvme1n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-nvme.1d0f-
4157533130333832453544373434313439344543-
416d617a6f6e20454332204e564d6520496e7374616e63652053746f72616765-00000001 ->
../../nvme0n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-nvme.1d0f-
4157533630333832453544373434313439344543-
416d617a6f6e20454332204e564d6520496e7374616e63652053746f72616765-00000001 ->
../../nvme1n1
```

In this example, all the OpenShift Container Storage worker nodes are of the Amazon EC2 type **i3.2xlarge**. So, all the three worker nodes have the same type of machine but the **by-id** identifier is unique for every local device. The **lsblk** command shows the last two devices, **nvme0n1** and **nvme1n1** with a size of 1.7 TB each.

For each worker node that has the OpenShift Container Storage label (a minimum of three), you need to find the unique **by-id** device names. In this example, the **by-id** device names are:

- **nvme-Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC**
- **nvme-Amazon_EC2_NVMe_Instance_Storage_AWS60382E5D7441494EC**



NOTE

You must repeat finding the device name **by-id** for all the other nodes that have the storage devices to be used by OpenShift Container Storage. See https://access.redhat.com/solutions/4928841 for more details.

1.2.5. Creating OpenShift Container Storage cluster on Amazon EC2 storage optimized - i3en.2xlarge instance type

Use this procedure to create OpenShift Container Storage cluster on Amazon EC2 (storage optimized - i3en.2xlarge instance type) infrastructure, which involves:

1. Creating PVs by using the **LocalVolume** CR
2. Creating a new **StorageClass**

The Amazon EC2 storage optimized - i3.2xlarge instance type includes two non-volatile memory express (NVMe) disks. The example in this procedure illustrates the use of both the disks that the instance type comes with.

WARNING

It is not recommended to use ephemeral storage of Amazon EC2 for OpenShift Container Storage persistent data, because stopping all the three nodes can cause data loss. It is recommended to use ephemeral storage only in scenarios such as the following:

- Cloud burst where data is copied from another location for a specific data crunching, which is limited in time
- Development or testing environment

When you are using the ephemeral storage of Amazon EC2, it is recommended to:

- Use three availability zones to decrease the risk of losing all the data
- Limit the number of users with **ec2:StopInstances** permissions to avoid instance shutdown by mistake

Prerequisites

- Ensure that all the requirements in the Requirements for installing OpenShift Container Storage using local storage devices section are met.
- Verify your OpenShift Container Platform worker nodes are labeled for OpenShift Container Storage, which is used as the **nodeSelector**.

```
$ oc get nodes -l cluster.ocs.openshift.io/openshift-storage -o jsonpath='{range .items[*]}
{.metadata.name}{"\n"}'
```

Example output:

```
ip-10-0-135-71.us-east-2.compute.internal
ip-10-0-145-125.us-east-2.compute.internal
ip-10-0-160-91.us-east-2.compute.internal
```

Procedure

1. Create local persistent volumes (PVs) on the storage nodes using **LocalVolume** custom resource (CR).
   Example of **LocalVolume** CR **local-storage-block.yaml** using OpenShift Storage Container label as node selector and **by-id** device identifier:

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: local-storage
  labels:
    app: ocs-storagecluster
spec:
  nodeSelector:
    nodeSelectorTerms:
      - matchExpressions:
          - key: cluster.ocs.openshift.io/openshift-storage
            operator: In
            values:
              - ''
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC   # <-- modify this line
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS1F45C01D7E84FE3E9   # <-- modify this line
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS136BC945B4ECB9AE4   # <-- modify this line
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441464EP   # <-- modify this line
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS1F45C01D7E84F43E7   # <-- modify this line
        - /dev/disk/by-id/nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS136BC945B4ECB9AE8   # <-- modify this line
```

   Each Amazon EC2 instance has two disks and this example uses both the disks.

2. Create the **LocalVolume** CR.

```
$ oc create -f local-storage-block.yaml
```

   Example output:

```
localvolume.local.storage.openshift.io/local-block created
```

3. Check if the pods are created.

```
$ oc -n local-storage get pods
```

Example output:

```
NAME                                  READY   STATUS    RESTARTS   AGE
local-block-local-diskmaker-59rmn     1/1     Running   0          15m
local-block-local-diskmaker-6n7ct     1/1     Running   0          15m
local-block-local-diskmaker-jwtsn     1/1     Running   0          15m
local-block-local-provisioner-6ssxc   1/1     Running   0          15m
local-block-local-provisioner-swwvx   1/1     Running   0          15m
local-block-local-provisioner-zmv5j   1/1     Running   0          15m
local-storage-operator-7848bbd595-686dg   1/1     Running   0          15m
```

4. Check if the PVs are created.
   You must see a new PV for each of the local storage devices on the three worker nodes. Refer the example in the Finding available storage devices section that shows two available storage devices per worker node with a size 1.7 TB for each node.

```
$ oc get pv
```

Example output:

```
NAME                CAPACITY ACCESS MODES  RECLAIM POLICY STATUS     CLAIM
STORAGECLASS  REASON   AGE
local-pv-1a46bc79  2328Gi   RWO          Delete         Available         localblock           14m
local-pv-429d90ee  2328Gi   RWO          Delete         Available         localblock           14m
local-pv-4d0a62e3  2328Gi   RWO          Delete         Available         localblock           14m
local-pv-55c05d76  2328Gi   RWO          Delete         Available         localblock           14m
local-pv-5c7b0990  2328Gi   RWO          Delete         Available         localblock           14m
local-pv-a6b283b   2328Gi   RWO          Delete         Available         localblock           14m
```

5. Check if a new **StorageClass** is created due to the creation of **LocalVolume** CR. This **StorageClass** is used while creating **StorageCluster** to create PVCs.

```
$ oc get sc | grep localblock
```

Example output:

```
NAME          PROVISIONER                    AGE
localblock    kubernetes.io/no-provisioner   7m46s
```

6. Create **StorageCluster** CR that uses the **localblock StorageClass** and the three PVs that are created.
   Example **StorageCluster** CR **ocs-cluster-service.yaml** using **monDataDirHostPath** and **localblock** StorageClass.

```
apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
```

```
resources:
  mds:
    limits:
      cpu: 3
    requests:
      cpu: 1
  noobaa-core:
    limits:
      cpu: 2
      memory: 8Gi
    requests:
      cpu: 1
      memory: 8Gi
  noobaa-db:
    limits:
      cpu: 2
      memory: 8Gi
    requests:
      cpu: 1
      memory: 8Gi
monDataDirHostPath: /var/lib/rook
storageDeviceSets:
  - count: 2
    dataPVCTemplate:
      spec:
        accessModes:
          - ReadWriteOnce
        resources:
          requests:
            storage: 2328Gi
        storageClassName: localblock
        volumeMode: Block
    name: ocs-deviceset
    placement: {}
    portable: false
    replica: 3
    resources: {}
```

IMPORTANT

To ensure that the OSDs have a guaranteed size across the nodes, the storage size for **storageDeviceSets** must be specified as less than or equal to the size of the desired PVs created on the nodes.

7. Create **StorageCluster** CR.

```
$ oc create -f ocs-cluster-service.yaml
```

Example output

```
storagecluster.ocs.openshift.io/ocs-cluster-service created
```

Verification steps

See Verifying your OpenShift Container Storage installation .

1.2.6. Creating OpenShift Container Storage cluster on VMware

Use this procedure to create storage cluster on VMware infrastructure.

VMware supports the following three types of local storage:

- Virtual machine disk (VMDK)
- Raw device mapping (RDM)
- VMDirectPath I/O

Prerequisites

- Ensure that all the requirements in the Requirements for installing OpenShift Container Storage using local storage devices section are met.
- You must have three worker nodes with the same storage size and type attached to each node to use local storage devices on VMware.
- Verify your OpenShift Container Platform worker nodes are labeled for OpenShift Container Storage:

```
$ oc get nodes -l cluster.ocs.openshift.io/openshift-storage -o jsonpath='{range .items[*]}
{.metadata.name}{"\n"}'
```

To identify storage devices on each node, refer to Finding available storage devices.

Procedure

1. Create the LocalVolume CR for block PVs.
   Example of **LocalVolume** CR **local-storage-block.yaml** using OpenShift Container Storage label as node:

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: local-storage
  labels:
    app: ocs-storagecluster
spec:
  nodeSelector:
    nodeSelectorTerms:
    - matchExpressions:
        - key: cluster.ocs.openshift.io/openshift-storage
          operator: In
          values:
          - ""
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
```

```
    - /dev/disk/by-id/scsi-36000c2991c27c2e5ba7c47d1e4352de2   # <-- modify this line
    - /dev/disk/by-id/scsi-36000c29682ca9e347926406711f3dc4e   # <-- modify this line
    - /dev/disk/by-id/scsi-36000c296aaf03a9b1e4b01d086bc6348   # <-- modify this line
```

2. Create **LocalVolume** CR for block PVs.

```
$ oc create -f local-storage-block.yaml
```

Example output:

```
localvolume.local.storage.openshift.io/local-block created
```

3. Check if the pods are created.

```
$ oc -n local-storage get pods
```

Example output:

```
NAME                                  READY   STATUS    RESTARTS   AGE
local-block-local-diskmaker-5brzv         1/1     Running   0          31s
local-block-local-diskmaker-8sxcs         1/1     Running   0          31s
local-block-local-diskmaker-s7s9p         1/1     Running   0          31s
local-block-local-provisioner-9cbw8       1/1     Running   0          31s
local-block-local-provisioner-cpddv       1/1     Running   0          31s
local-block-local-provisioner-f6h7h       1/1     Running   0          31s
local-storage-operator-75b9776b75-vwdzh   1/1     Running   0          2m47s
```

4. Check the new **localblock** StorageClass.

```
$ oc get sc | grep localblock
```

Example output:

```
NAME          PROVISIONER                    AGE
localblock    kubernetes.io/no-provisioner   8m38s
```

5. Check the PVs that are created with the **Available** status.

```
$ oc get pv
```

Example output:

```
NAME                CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM
STORAGECLASS   REASON   AGE
local-pv-150fdc87   100Gi      RWO            Delete           Available           localblock
2m11s
local-pv-183bfc0a   100Gi      RWO            Delete           Available           localblock
2m11s
local-pv-b2f5cb25   100Gi      RWO            Delete           Available           localblock
2m21s
```

In this example, three PVs are used for OSD storage (100 GB).

6. Create **StorageCluster** CR **ocs-cluster-service-VMware.yaml** that uses the **monDataDirHostPath** and **localblock** StorageClass.

```
apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
  monDataDirHostPath: /var/lib/rook
  storageDeviceSets:
  - count: 1
    dataPVCTemplate:
      spec:
        accessModes:
        - ReadWriteOnce
        resources:
          requests:
            storage: 100Gi
        storageClassName: localblock
        volumeMode: Block
    name: ocs-deviceset
    placement: {}
    portable: false
    replica: 3
    resources: {}
```

IMPORTANT

To ensure that the OSDs have a guaranteed size across the nodes, the storage size for **storageDeviceSets** must be specified as less than or equal to the size of the desired PVs created on the nodes.

7. Create **StorageCluster** CR.

```
$ oc create -f ocs-cluster-service-VMware.yaml
```

Example output:

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

Verification steps

See Verifying your OpenShift Container Storage installation .

1.2.7. Creating OpenShift Container Storage cluster on bare metal

Prerequisites

- Ensure that all the requirements in the Requirements for installing OpenShift Container Storage using local storage devices section are met.

- You must have three worker nodes with the same storage size and type attached to each node (for example, 2TB NVMe hard drive) to use local storage devices on bare metal.
- Verify your OpenShift Container Platform worker nodes are labeled for OpenShift Container Storage:

```
$ oc get nodes -l cluster.ocs.openshift.io/openshift-storage -o jsonpath='{range .items[*]}
{.metadata.name}{"\n"}'
```

To identify storage devices on each node, refer to Finding available storage devices.

Procedure

1. Create **LocalVolume** CR for block PVs.
   Example of **LocalVolume** CR **local-storage-block.yaml** using OCS label as node selector.

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: local-storage
  labels:
    app: ocs-storagecluster
spec:
  nodeSelector:
    nodeSelectorTerms:
    - matchExpressions:
        - key: cluster.ocs.openshift.io/openshift-storage
          operator: In
          values:
          - ""
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY81260978128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY80440W5U128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPYB85AABDE128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY0A60CB81128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY0093D45E128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPYE46F6060128A   # <-- modify
this line
```

2. Create the **LocalVolume** CR for block PVs.

```
$ oc create -f local-storage-block.yaml
```

3. Check if the pods are created.

```
$ oc -n local-storage get pods
```

4. Check if the PVs are created.

```
$ oc get pv
```

5. Check for the new **localblock StorageClass**.

```
$ oc get sc | grep localblock
```

Example output:

```
NAME          PROVISIONER                    AGE
localblock    kubernetes.io/no-provisioner   10m20s
```

6. Create **StorageCluster** CR **cluster-service-metal.yaml** using **monDataDirHostPath** and **localblock** storage classes.

```
apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
  monDataDirHostPath: /var/lib/rook
  storageDeviceSets:
  - count: 2
    dataPVCTemplate:
      spec:
        accessModes:
        - ReadWriteOnce
        resources:
          requests:
            storage: 2Ti
        storageClassName: localblock
        volumeMode: Block
    name: ocs-deviceset
    placement: {}
    portable: false
    replica: 3
    resources: {}
```

IMPORTANT

To ensure that the OSDs have a guaranteed size across the nodes, the storage size for **storageDeviceSets** must be specified as less than or equal to the size of the desired PVs created on the nodes.

7. Create the **StorageCluster** CR.

```
$ oc create -f cluster-service-metal.yaml
```

Example output:

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

Verification steps

See Verifying your OpenShift Container Storage installation .

CHAPTER 2. VERIFYING YOUR OPENSHIFT CONTAINER STORAGE INSTALLATION

Use this section to verify that OpenShift Container Storage is deployed correctly.

2.1. VERIFY THAT THE PODS ARE IN RUNNING STATE

- Click **Workloads → Pods** from the left pane of the OpenShift Web Console.
- Select **openshift-storage** from the **Project** drop down list.
  The number of pods varies depending on the OSDs and number of worker nodes deployed on OpenShift Container Platform. Number of OSDs depend on **Count** and **Replica** defined for each **StorageDeviceSet** in **StorageCluster**. The number of pods per component is not directly related only to the number of worker nodes but to OSDs as well. This should be explained better: Which pods are expected to have such depending on number of worker nodes, which ones on OSDs (or sometimes capacity added when not using local storage), and which ones should have exact number as listed.

  Verify that the following pods are in running and completed state by clicking on the **Running** and the **Completed** tabs:

Table 2.1. Pods corresponding to storage components for a three worker node cluster

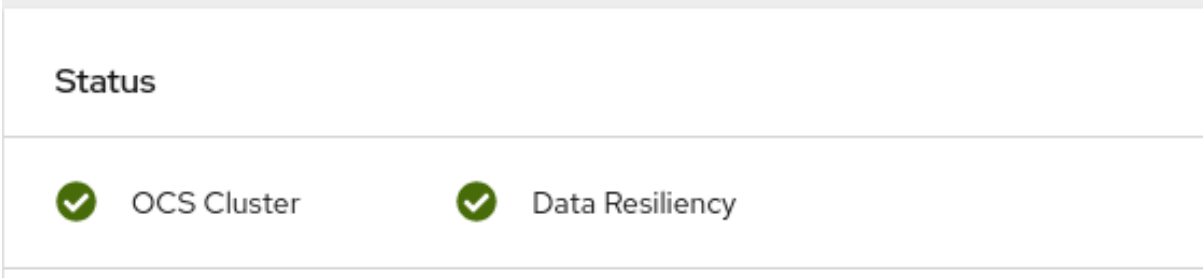
Component	No. of pods	Name of the pod
Number of pods that you must see for the following components:		
OpenShift Container Storage Operator	1	**ocs-operator-***
Rook-ceph Operator	1	**rook-ceph-operator-***
NooBaa	4	○ **noobaa-operator-*** ○ **noobaa-core-*** ○ **nooba-db-*** ○ **noobaa-endpoint-***
Mon	3	○ **rook-ceph-mon-*** ○ **rook-ceph-mon-*** ○ **rook-ceph-mon-*** (on different nodes)
rook-ceph-mgr	1	**rook-ceph-mgr-*** (on storage node)

Component	No. of pods	Name of the pod
MDS	2	**rook-ceph-mds-ocs-storagecluster-cephfilesystem-*** (2 pods on different storage nodes)
lib-bucket-provisioner	1	**lib-bucket-provisioner--*** (on any node)
Number of pods for CSI vary depending on the number of nodes selected as storage nodes (a minimum of 3 nodes)		
CSI	10	○ **cephfs** (at least 5 pods) ■ **csi-cephfsplugin-*** (1 on each node where storage is consumed, that is, 3 pods on different nodes) ■ **csi-cephfsplugin-provisioner-*** (2 pods on different storage nodes if available) ○ **rbd** (at least 5 pods in total) ■ **csi-rbdplugin-*** (one on each node where storage is consumed, that is, 3 pods on different nodes) ■ **csi-rbdplugin-provisioner-*** (2 pods on different storage nodes if available)
rook-ceph-drain-canary	3	**rook-ceph-drain-canary-*** (3 pods, that is, one on each storage node)
rook-ceph-crashcollector	3	**rook-ceph-crashcollector-*** (3 pods)
Number of OSDs vary depending on **Count** and **Replica** defined for each StorageDeviceSet in StorageCluster.		
OSD	6	○ **rook-ceph-osd-*** (3 pods on different nodes) ○ **rook-ceph-osd-prepare-ocs-deviceset-*** (3 pods)

2.2. VERIFY THAT THE OPENSHIFT CONTAINER STORAGE CLUSTER IS HEALTHY

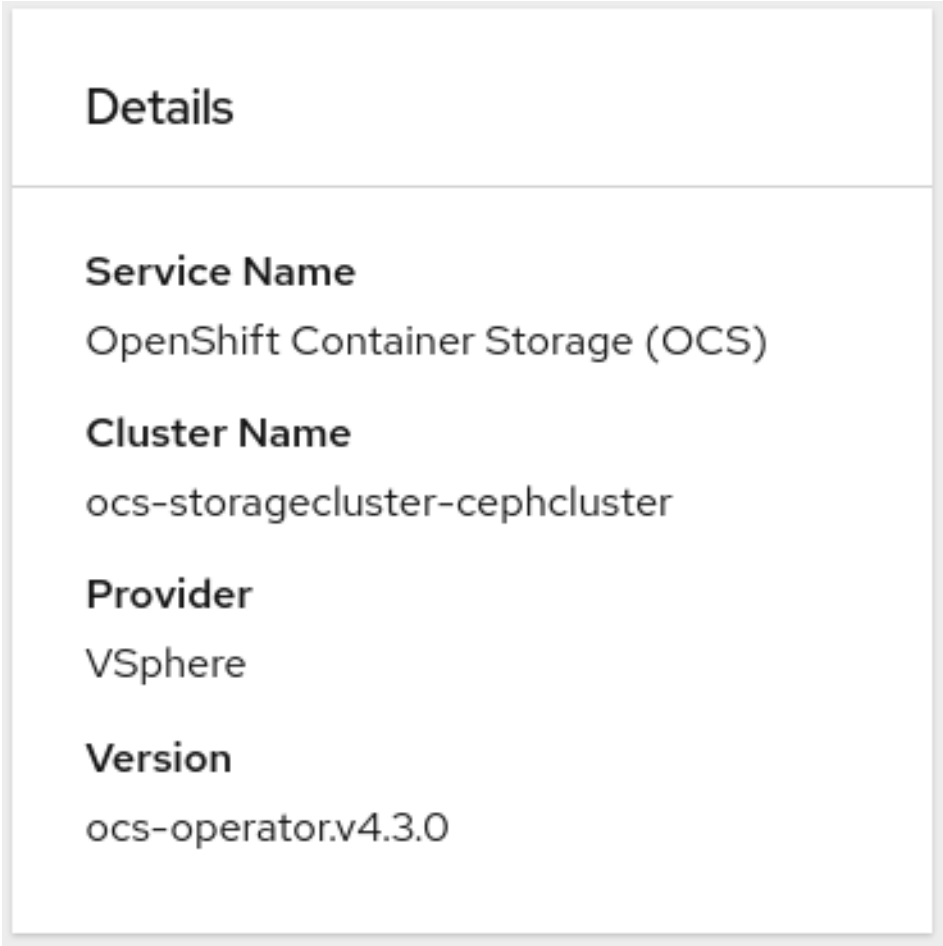
- Click **Home → Dashboards** from the left pane of the OpenShift Web Console and click **OCS PV** tab.
  In the **Status card**, verify that *OCS Cluster* has a green tick mark as shown in the following image:

Figure 2.1. Health status card in Persistent Storage (OCS PV) dashboard



In the **Details card**, verify that the cluster information is displayed appropriately as follows:

Figure 2.2. Details card in Persistent Storage (OCS PV) dashboard

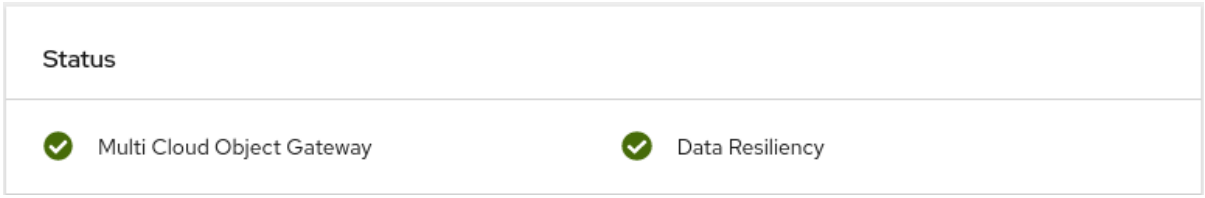


For more information on verifying the health of OpenShift Container Storage cluster using the persistent storage dashboard, see Monitoring OpenShift Container Storage .

2.3. VERIFY THAT THE MULTICLOUD OBJECT GATEWAY IS HEALTHY

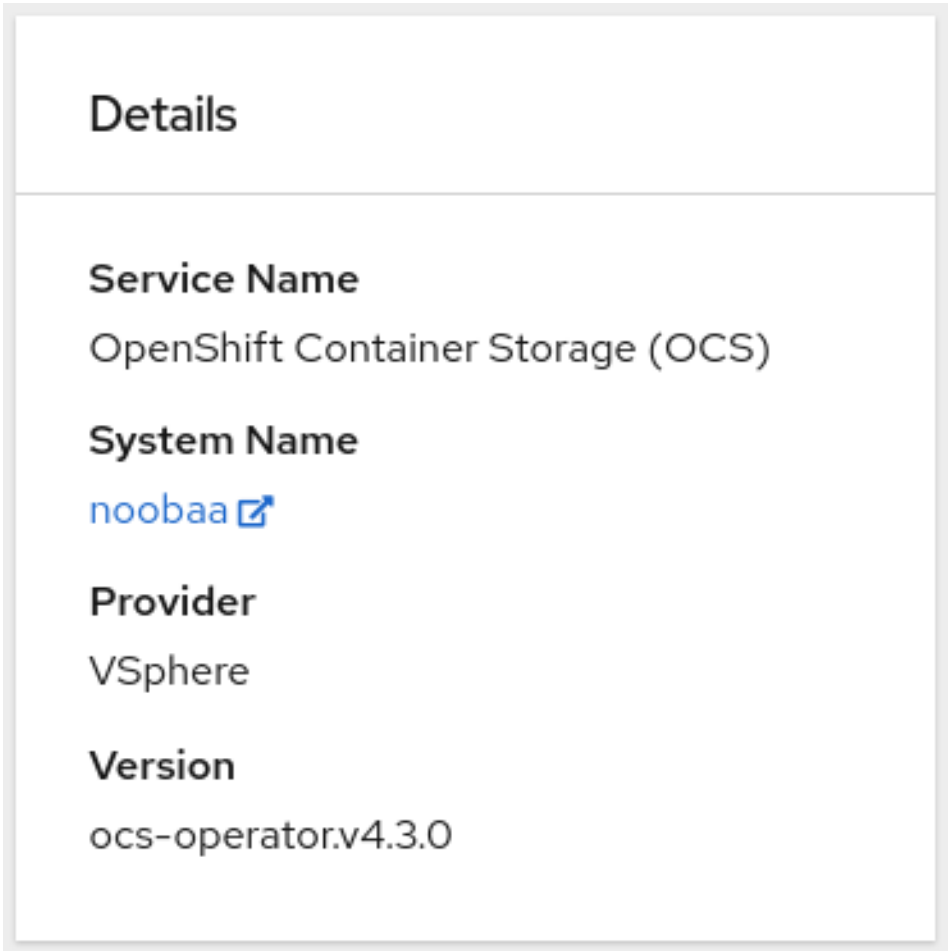
- Click **Home → Dashboards** from the left pane of the OpenShift Web Console and click the **OCS Object Service** tab.
  In the **Status card**, verify that the Multicloud Object Gateway (MCG) storage displays a green tick icon as shown in following image:

Figure 2.3. Health status card in Object Service dashboard



In the **Details card**, verify that the MCG information is displayed appropriately as follows:

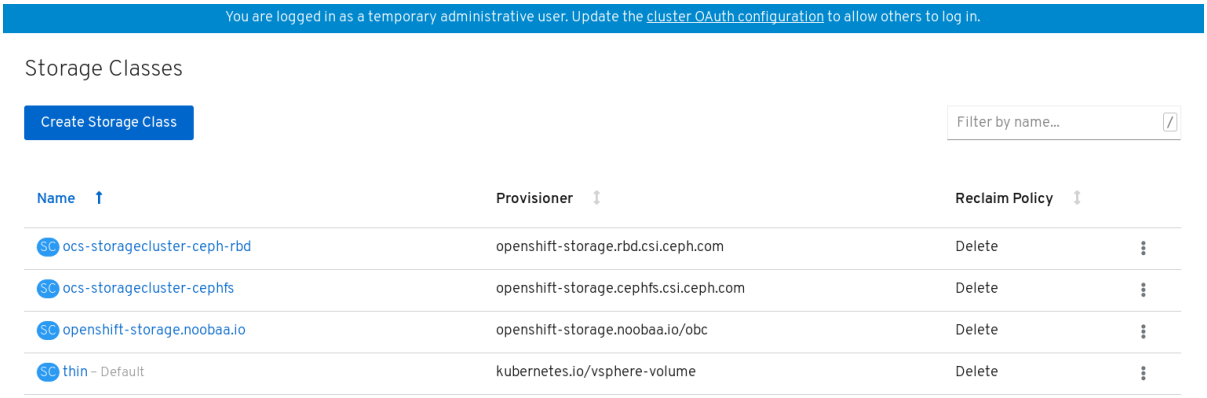
Figure 2.4. Details card in Object Service dashboard



For more information on verifying the health of OpenShift Container Storage cluster using the object service dashboard, see Monitoring OpenShift Container Storage .

2.4. VERIFY THAT THE STORAGE CLASSES ARE CREATED AND LISTED

- Click **Storage → Storage Classes**from the left pane of the OpenShift Web Console. Verify that the following three storage classes are created with the OpenShift Container Storage cluster creation:
  - **ocs-storagecluster-ceph-rbd**
  - **ocs-storagecluster-cephfs**
  - **openshift-storage.noobaa.io**


You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.
Storage Classes
Create Storage Class
Filter by name...
Name
Provisioner
Reclaim Policy
SC ocs-storagecluster-ceph-rbd
openshift-storage.rbd.csi.ceph.com
Delete
SC ocs-storagecluster-cephfs
openshift-storage.cephfs.csi.ceph.com
Delete
SC openshift-storage.noobaa.io
openshift-storage.noobaa.io/obc
Delete
SC thin - Default
kubernetes.io/vsphere-volume
Delete

CHAPTER 3. UNINSTALLING OPENSHIFT CONTAINER STORAGE

Use the steps in this section to uninstall OpenShift Container Storage instead of the **Uninstall** option from the user interface.

Prerequisites

- Make sure that the OpenShift Container Storage cluster is in healthy state. The deletion might fail if some of the pods are not terminated successfully due to insufficient resources or nodes. In case the cluster is in unhealthy state, you should contact Red Hat Customer Support before uninstalling OpenShift Container Storage.
- Delete any applications that are consuming persistent volume claims (PVCs) or object bucket claims (OBCs) based on the OpenShift Container Storage storage classes and then delete PVCs and OBCs that are using OpenShift Container Storage storage classes.

Procedure

1. List the storage classes and take a note of the storage classes with the following storage class provisioners:
   - **openshift-storage.rbd.csi.ceph.com**
   - **openshift-storage.cephfs.csi.ceph.com**
   - **openshift-storage.noobaa.io/obc**

   For example

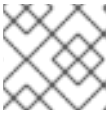
   ```
   $ oc get storageclasses
   NAME                          PROVISIONER                             AGE
   gp2 (default)                 kubernetes.io/aws-ebs                   23h
   ocs-storagecluster-ceph-rbd   openshift-storage.rbd.csi.ceph.com      22h
   ocs-storagecluster-cephfs     openshift-storage.cephfs.csi.ceph.com   22h
   openshift-storage.noobaa.io   openshift-storage.noobaa.io/obc         22h
   ```

2. Query for PVCs and OBCs that are using the storage class provisioners listed in the previous step.

   ```
   $ oc get pvc -o=jsonpath='{range .items[?(@.spec.storageClassName=="ocs-storagecluster-ceph-rbd")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{" Labels: "}{@.metadata.labels}{"\n"}{end}' --all-namespaces|awk '! ( /Namespace: openshift-storage/ && /app:noobaa/ )'
   ```

   ```
   $ oc get pvc -o=jsonpath='{range .items[?(@.spec.storageClassName=="ocs-storagecluster-cephfs")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
   ```

   ```
   $ oc get obc -o=jsonpath='{range .items[?(@.spec.storageClassName=="openshift-storage.noobaa.io")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
   ```



NOTE

Ignore any NooBaa PVCs in the **openshift-storage** namespace.

3. Follow these instructions to ensure that the PVCs listed in the previous step are deleted:

   a. Determine the pod that is consuming the PVC.

   b. Identify the controlling object such as a **Deployment**, **StatefulSet**, **DeamonSet**, **Job**, or a custom controller.
   Each object has a metadata field known as **OwnerReference**. This is a list of associated objects. The **OwnerReference** with the **controller** field set to **true** will point to controlling objects such as **ReplicaSet**, **StatefulSet**, **DaemonSet** and so on.

   c. Ensure that the object is safe to delete by asking the owner of the project and then delete it.

   d. Delete the PVCs and OBCs.

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

   If you have created any PVCs as a part of configuring the monitoring stack, cluster logging operator, or prometheus registry, then you must perform the clean up steps provided in the following sections as required:

   - Section 3.1, “Removing monitoring stack from OpenShift Container Storage”
   - Section 3.2, “Removing OpenShift Container Platform registry from OpenShift Container Storage”
   - Section 3.3, “Removing the cluster logging operator from OpenShift Container Storage”

4. List and note the backing local volume objects. If no results found, then skip step 8 & 9.

   ```
   for sc in $(oc get storageclass|grep 'kubernetes.io/no-provisioner' |grep -E $(oc get
   storagecluster -n openshift-storage -o jsonpath='{
   .items[*].spec.monPVCTemplate.spec.storageClassName } {
   .items[*].spec.storageDeviceSets[*].dataPVCTemplate.spec.storageClassName}' | sed 's/
   /|/g')| awk '{ print $1 }');
   do
      echo -n "StorageClass: $sc ";
      oc get storageclass $sc -o jsonpath=" { 'LocalVolume: ' }{
   .metadata.labels['local\.storage\.openshift\.io/owner-name'] } { '\n' }";
   done
   ```

   Example output

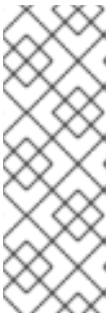
   ```
   StorageClass: localfile  LocalVolume: local-file
   StorageClass: localblock  LocalVolume: local-block
   ```

5. Delete the **StorageCluster** object.

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

6. Delete the namespace and wait till the deletion is complete.

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



NOTE

You will need to switch to another project if openshift-storage was the active project.

For example

```
$ oc project default
```

7. Clean up the storage operator artifacts on each node.

```
$ 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 /var/lib/rook; done
```

Ensure you see **removed directory '/var/lib/rook'** in the output.

Example output

```
Starting pod/ip-10-0-134-65us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
removed '/var/lib/rook/openshift-storage/log/ocs-deviceset-2-0-gk22s/ceph-volume.log'
removed directory '/var/lib/rook/openshift-storage/log/ocs-deviceset-2-0-gk22s'
removed '/var/lib/rook/openshift-storage/log/ceph-osd.2.log'
removed '/var/lib/rook/openshift-storage/log/ceph-volume.log'
removed directory '/var/lib/rook/openshift-storage/log'
removed directory '/var/lib/rook/openshift-storage/crash/posted'
removed directory '/var/lib/rook/openshift-storage/crash'
removed '/var/lib/rook/openshift-storage/client.admin.keyring'
removed '/var/lib/rook/openshift-storage/openshift-storage.config'
removed directory '/var/lib/rook/openshift-storage'
removed '/var/lib/rook/osd2/openshift-storage.config'
removed directory '/var/lib/rook/osd2'
removed directory '/var/lib/rook'

Removing debug pod ...
Starting pod/ip-10-0-155-149us-east-2computeinternal-debug ...
.
.
removed directory '/var/lib/rook'

Removing debug pod ...
Starting pod/ip-10-0-162-89us-east-2computeinternal-debug ...
.
.
removed directory '/var/lib/rook'

Removing debug pod ...
```

8. Delete the local volume created during the deployment and for each of the local volumes listed in step 4.
   For each of the local volumes, do the following:

   a. Set the variable **LV** to the name of the LocalVolume and variable **SC** to name of the StorageClass.

      For example

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

   b. List and note the devices to be cleaned up later.

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

      Example output

      ```
      /dev/disk/by-id/nvme-Amazon_Elastic_Block_Store_vol078f5cdde09efc165 /dev/disk/by-id/nvme-Amazon_Elastic_Block_Store_vol0defc1d5e2dd07f9e /dev/disk/by-id/nvme-Amazon_Elastic_Block_Store_vol0c8e82a3beeb7b7e5
      ```

   c. Delete the local volume resource.

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

   d. 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
      ```

   e. 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 ouptut

      ```
      Starting pod/ip-10-0-141-2us-east-2computeinternal-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/ip-10-0-144-55us-east-2computeinternal-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/ip-10-0-175-34us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'

Removing debug pod ...
```

9. Wipe the disks for each of the local volumes listed in step 4 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
   ip-10-0-134-65.us-east-2.compute.internal    Ready    worker   4h45m   v1.17.1
   ip-10-0-155-149.us-east-2.compute.internal   Ready    worker   4h46m   v1.17.1
   ip-10-0-162-89.us-east-2.compute.internal    Ready    worker   4h45m   v1.17.1
   ```

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

   ```
   $ oc debug node/ip-10-0-134-65.us-east-2.compute.internal
   Starting pod/ip-10-0-134-65us-east-2computeinternal-debug ...
   To use host binaries, run `chroot /host`
   Pod IP: 10.0.134.65
   If you don't see a command prompt, try pressing enter.
   sh-4.2# chroot /host
   ```

   c. Store the disk paths gathered in step 8(ii) in the **DISKS** variable within quotes.

   ```
   sh-4.2# DISKS="/dev/disk/by-id/nvme-
   Amazon_Elastic_Block_Store_vol078f5cdde09efc165 /dev/disk/by-id/nvme-
   Amazon_Elasti_Block_Store_vol0defc1d5e2dd07f9e /dev/disk/by-id/nvme-
   Amazon_Elastic_Block_Store_vol0c8e82a3beeb7b7e5"
   ```

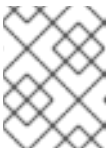
   d. Run **sgdisk --zap-all** on all the disks:

   ```
   sh-4.3# for disk in $DISKS; do sgdisk --zap-all $disk;done
   ```

   Example output

   ```
   Problem opening /dev/disk/by-id/nvme-
   Amazon_Elastic_Block_Store_vol078f5cdde09efc165 for reading! Error is 2.
   The specified file does not exist!
   Problem opening '' for writing! Program will now terminate.
   Warning! MBR not overwritten! Error is 2!
   Problem opening /dev/disk/by-id/nvme-
   Amazon_Elasti_Block_Store_vol0defc1d5e2dd07f9e for reading! Error is 2.
   The specified file does not exist!
   Problem opening '' for writing! Program will now terminate.
   Warning! MBR not overwritten! Error is 2!
   ```

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



NOTE

Ignore file-not-found warnings as they refer to disks that are on other machines.

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

```
sh-4.3# exit
exit
sh-4.2# exit
exit

Removing debug pod ...
```

10. Delete the storage classes with an **openshift-storage** provisioner listed in step 1.

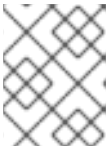
```
$ oc delete storageclass <storageclass-name> --wait=true --timeout=5m
```

For example:

```
$ oc delete storageclass ocs-storagecluster-ceph-rbd ocs-storagecluster-cephfs openshift-
storage.noobaa.io --wait=true --timeout=5m
```

11. Unlabel the storage nodes.

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



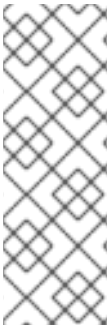
NOTE

You can ignore the warnings displayed for the unlabeled nodes such as **label <label> not found**.

12. Remove **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
storageclusterinitializations.ocs.openshift.io storageclusters.ocs.openshift.io  --wait=true --
timeout=5m
```

13. To make sure that OpenShift Container Storage is uninstalled, verify that the openshift-storage namespace no longer exists and the storage dashboard no longer appears in the UI.



NOTE

While uninstalling OpenShift Container Storage, if namespace is not deleted completely and remains in **Terminating** state, perform the steps in the article https://access.redhat.com/solutions/3881901 to identify objects that are blocking the namespace from being terminated. OpenShift objects such as **Cephcluster**, **StorageCluster**, **NooBaa**, and **PVC** that have the finalizers might be the cause for the namespace to be in **Terminating** state. If PVC has a finalizer, force delete the associated pod to remove the finalizer.

3.1. REMOVING MONITORING STACK FROM OPENSHIFT CONTAINER STORAGE

Use this section to clean up monitoring stack from OpenShift Container Storage.

The PVCs that are created as a part of configuring the monitoring stack are in the **openshift-monitoring** namespace.

Prerequisites

- PVCs are configured to use OpenShift Container Platform monitoring stack.
  For 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
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
```

3. Remove any **config** sections that reference the OpenShift Container Storage 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 Container Storage PVCs.

4. List the pods consuming the PVC.
In this example, the **alertmanagerMain** and **prometheusK8s** pods that were consuming the PVCs are in the **Terminating** state. You can delete the PVCs once these pods are no longer using OpenShift Container Storage PVC.

```
$ oc get pod,pvc -n openshift-monitoring
NAME                                          READY   STATUS      RESTARTS AGE
pod/alertmanager-main-0                       3/3     Terminating 0      10h
pod/alertmanager-main-1                       3/3     Terminating 0      10h
pod/alertmanager-main-2                       3/3     Terminating 0      10h
pod/cluster-monitoring-operator-84cd9df668-zhjfn  1/1     Running     0      18h
pod/grafana-5db6fd97f8-pmtbf                  2/2     Running     0      10h
pod/kube-state-metrics-895899678-z2r9q        3/3     Running     0      10h
pod/node-exporter-4njxv                       2/2     Running     0      18h
pod/node-exporter-b8ckz                       2/2     Running     0      11h
pod/node-exporter-c2vp5                       2/2     Running     0      18h
pod/node-exporter-cq65n                       2/2     Running     0      18h
pod/node-exporter-f5sm7                       2/2     Running     0      11h
pod/node-exporter-f852c                       2/2     Running     0      18h
pod/node-exporter-l9zn7                       2/2     Running     0      11h
pod/node-exporter-ngbs8                       2/2     Running     0      18h
pod/node-exporter-rv4v9                       2/2     Running     0      18h
pod/openshift-state-metrics-77d5f699d8-69q5x  3/3     Running     0      10h
pod/prometheus-adapter-765465b56-4tbxx        1/1     Running     0      10h
pod/prometheus-adapter-765465b56-s2qg2        1/1     Running     0      10h
pod/prometheus-k8s-0                          6/6     Terminating 1      9m47s
pod/prometheus-k8s-1                          6/6     Terminating 1      9m47s
pod/prometheus-operator-cbfd89f9-ldnwc        1/1     Running     0      43m
pod/telemeter-client-7b5ddb4489-2xfpz         3/3     Running     0      10h

NAME                                                STATUS  VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS                AGE
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-0   Bound    pvc-
2eb79797-1fed-11ea-93e1-0a88476a6a64   40Gi       RWO            ocs-storagecluster-ceph-
```

```
rbd   19h
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-1   Bound   pvc-
2ebeee54-1fed-11ea-93e1-0a88476a6a64   40Gi       RWO           ocs-storagecluster-ceph-
rbd   19h
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-2   Bound   pvc-2ec6a9cf-
1fed-11ea-93e1-0a88476a6a64   40Gi       RWO           ocs-storagecluster-ceph-rbd   19h
persistentvolumeclaim/ocs-prometheus-claim-prometheus-k8s-0        Bound   pvc-3162a80c-
1fed-11ea-93e1-0a88476a6a64   40Gi       RWO           ocs-storagecluster-ceph-rbd   19h
persistentvolumeclaim/ocs-prometheus-claim-prometheus-k8s-1        Bound   pvc-
316e99e2-1fed-11ea-93e1-0a88476a6a64   40Gi       RWO           ocs-storagecluster-ceph-
rbd   19h
```

5. Delete 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
```

3.2. REMOVING OPENSHIFT CONTAINER PLATFORM REGISTRY FROM OPENSHIFT CONTAINER STORAGE

Use this section to clean up OpenShift Container Platform registry from OpenShift Container Storage. If you want to configure an alternative storage, see: https://access.redhat.com/documentation/en-us/openshift_container_platform/4.3/html-single/registry/architecture-component-imageregistry

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

Prerequisites

- The image registry should have been configured to use an OpenShift Container Storage 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
```

- For AWS:

Before editing

```
.
.
storage:
  pvc:
    claim: registry-cephfs-rwx-pvc
.
.
```

After editing

```
.
.
storage:
.
.
```

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

- For VMware:

Before editing

```
.
.
storage:
  pvc:
    claim: registry-cephfs-rwx-pvc
.
.
```

After editing

```
.
.
storage:
  emptyDir: {}
.
.
```

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
```

3.3. REMOVING THE CLUSTER LOGGING OPERATOR FROM OPENSHIFT CONTAINER STORAGE

Use this section to clean up the cluster logging operator from OpenShift Container Storage.

The PVCs that are created as a part of configuring cluster logging operator are in **openshift-logging** namespace.

Prerequisites

- The cluster logging instance should have been configured to use OpenShift Container Storage 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 PVCs.

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