Red Hat OpenShift Container Storage 4.2

Managing OpenShift Container Storage

Instructions for cluster and storage administrators

Last Updated: 2020-06-03

Red Hat OpenShift Container Storage 4.2 Managing OpenShift Container Storage

Instructions for cluster and storage administrators

Abstract

This document covers instructions for managing an OpenShift Container Storage cluster.

Table of Contents

CHAPTER 1. OVERVIEW

Managing OpenShift Container Storage is written to help administrators understand how to manage and administer their Red Hat OpenShift Container Storage cluster.

Most management tasks focus on a single resource. This document is divided into chapters based on the resource that an administrator is trying to modify:

- Chapter 2, *Configure storage for OpenShift Container Platform services* shows you how to use OpenShift Container Storage for core OpenShift Container Platform services.
- Chapter 3, *Backing OpenShift Container Platform applications with OpenShift Container Storage* provides information about how to configure OpenShift Container Platform applications to use OpenShift Container Storage.
- Chapter 4, *Scaling storage nodes* provides information about scaling storage capacity of OpenShift Container Storage nodes.
- Chapter 5, *Managing persistent volume claims* provides information about managing Persistent Volume Claim requests, and automating the fulfillment of those requests.
- Chapter 6, *Multicloud Object Gateway* provides information about the Multicloud Object Gateway.
- Chapter 7, *Replacing storage nodes for OpenShift Container Storage* shows you how to replace an operational or failed node on AWS UPI, AWS IPI, and VMware UPI for OpenShift Container Storage.
- Chapter 8, *Updating OpenShift Container Storage* provides instructions for upgrading your OpenShift Container Storage cluster.

CHAPTER 2. CONFIGURE STORAGE FOR OPENSHIFT CONTAINER PLATFORM SERVICES

You can use OpenShift Container Storage to provide storage for OpenShift Container Platform services such as image registry, monitoring, and logging.

The process for configuring storage for these services depends on the infrastructure used in your OpenShift Container Storage deployment.

WARNING

Always ensure that you have plenty of storage capacity for these services. If the storage for these critical services runs out of space, the cluster becomes inoperable and very difficult to recover.

Red Hat recommends configuring shorter curation and retention intervals for these services. See Configuring Curator and Modifying retention time for Prometheus metrics data in the OpenShift Container Platform documentation for details.

If you do run out of storage space for these services, contact Red Hat Customer Support.

2.1. CONFIGURING IMAGE REGISTRY TO USE OPENSHIFT CONTAINER STORAGE

OpenShift Container Platform provides a built in Container Image Registry which runs as a standard workload on the cluster. A registry is typically used as a publication target for images built on the cluster as well as a source of images for workloads running on the cluster.

Follow the instructions in this section to configure OpenShift Container Storage as storage for the Container Image Registry. On AWS, it is not required to change the storage for the registry. However, it is recommended to change the storage to OpenShift Container Storage persistent volume for vSphere platform.

WARNING

This process does not migrate data from an existing image registry to the new image registry. If you already have container images in your existing registry, back up your registry before you complete this process, and re-register your images when this process is complete.

Prerequisites

- You have administrative access to OpenShift Web Console.

- OpenShift Container Storage Operator is installed and running in the **openshift-storage** namespace. In OpenShift Web Console, click **Operators → Installed Operators** to view installed operators.
- Image Registry Operator is installed and running in the **openshift-image-registry** namespace. In OpenShift Web Console, click **Administration → Cluster Settings → Cluster Operators** to view cluster operators.
- The **ocs-storagecluster-cephfs** storage class is available. In OpenShift Web Console, click **Storage → Storage Classes** to view available storage classes.

Procedure

1. **Create a Persistent Volume Claim for the Image Registry to use.**
   a. In OpenShift Web Console, click **Storage → Persistent Volume Claims**.
   b. Set the **Project** to **openshift-image-registry**.
   c. Click **Create Persistent Volume Claim**
      i. Specify a **Storage Class** of **ocs-storagecluster-cephfs**.
      ii. Specify the Persistent Volume Claim **Name**, for example, **ocs4registry**.
      iii. Specify an **Access Mode** of **Shared Access (RWX)**.
      iv. Specify a **Size** of at least 100 GB.
      v. Click **Create**.
      Wait until the status of the new Persistent Volume Claim is listed as **Bound**.
2. **Configure the cluster's Image Registry to use the new Persistent Volume Claim.**
   a. Click **Administration →Custom Resource Definitions**.
   b. Click the **Config** custom resource definition associated with the **imageregistry.operator.openshift.io** group.
   c. Click the **Instances** tab.
   d. Beside the cluster instance, click the **Action Menu (⋮) → Edit Config**.
   e. Add the new Persistent Volume Claim as persistent storage for the Image Registry.
      i. Add the following under **spec:**, replacing the existing **storage:** section if necessary.

```
storage:
  pvc:
    claim: <new-pvc-name>
```

For example:

```
storage:
  pvc:
    claim: ocs4registry
```

ii. Click **Save**.

3. **Verify that the new configuration is being used.**

   a. Click **Workloads** → **Pods**.

   b. Set the **Project** to **openshift-image-registry**.

   c. Verify that the new **image-registry-*** pod appears with a status of **Running**, and that the previous **image-registry-*** pod terminates.

   d. Click the new **image-registry-*** pod to view pod details.

   e. Scroll down to **Volumes** and verify that the **registry-storage** volume has a **Type** that matches your new Persistent Volume Claim, for example, **ocs4registry**.

2.2. CONFIGURING MONITORING TO USE OPENSHIFT CONTAINER STORAGE

OpenShift Container Storage provides a monitoring stack that is comprised of Prometheus and AlertManager.

Follow the instructions in this section to configure OpenShift Container Storage as storage for the monitoring stack.

IMPORTANT

Monitoring will not function if it runs out of storage space. Always ensure that you have plenty of storage capacity for monitoring.

Red Hat recommends configuring a short retention intervals for this service. See the *Modifying retention time for Prometheus metrics data* sub section of Configuring persistent storage in the OpenShift Container Platform documentation for details.

Prerequisites

- You have administrative access to OpenShift Web Console.

- OpenShift Container Storage Operator is installed and running in the **openshift-storage** namespace. In OpenShift Web Console, click **Operators** → **Installed Operators** to view installed operators.

- Monitoring Operator is installed and running in the **openshift-monitoring** namespace. In OpenShift Web Console, click **Administration** → **Cluster Settings** → **Cluster Operators** to view cluster operators.

- The **ocs-storagecluster-ceph-rbd** storage class is available. In OpenShift Web Console, click **Storage** → **Storage Classes** to view available storage classes.

Procedure

1. In OpenShift Web Console, go to **Workloads** → **Config Maps**.

2. Set the **Project** dropdown to **openshift-monitoring**.

3. Click **Create Config Map**.

4. Define a new **openshift-monitoring-config** Config Map using the following example.
   Replace the content in angle brackets (**<**, **>**) with your own values, for example, **retention: 24h** or **storage: 40Gi**.

   Example openshift-monitoring-config Config Map

   ```
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: cluster-monitoring-config
     namespace: openshift-monitoring
   data:
     config.yaml: |
         prometheusK8s:
           retention: <time to retain monitoring files, e.g. 24h>
           volumeClaimTemplate:
             metadata:
               name: ocs-prometheus-claim
             spec:
               storageClassName: ocs-storagecluster-ceph-rbd
               resources:
                 requests:
                   storage: <size of claim, e.g. 40Gi>
         alertmanagerMain:
           volumeClaimTemplate:
             metadata:
               name: ocs-alertmanager-claim
             spec:
               storageClassName: ocs-storagecluster-ceph-rbd
               resources:
                 requests:
                   storage: <size of claim, e.g. 40Gi>
   ```

5. Click **Create** to save and create the Config Map.

Verification steps

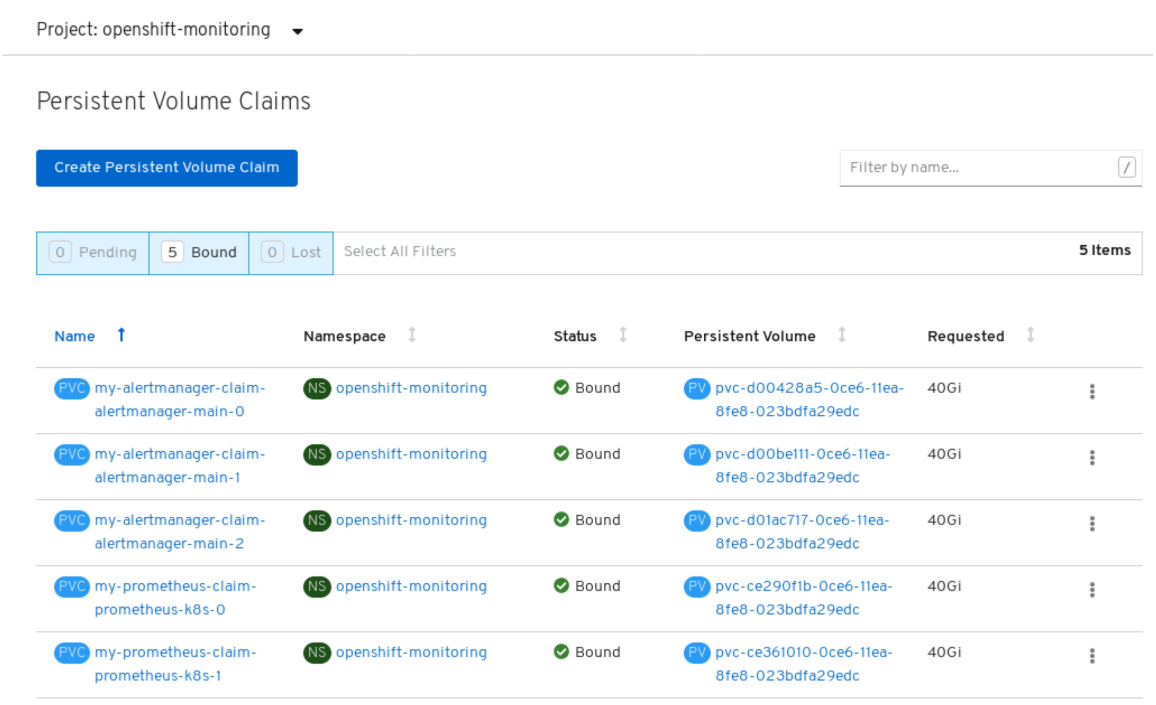
1. Verify that the Persistent Volume claims are bound to the pods.

   a. Go to **Storage → Persistent Volume Claims**.

   b. Set the **Project** dropdown to **openshift-monitoring**.

   c. Verify that 5 Persistent Volume Claims are visible with a state of **Bound**, attached to three **alertmanager-main-*** pods, and two **prometheus-k8s-*** pods.

      Monitoring storage created and bound


Project: openshift-monitoring

Persistent Volume Claims

Create Persistent Volume Claim

Filter by name...

0 Pending | 5 Bound | 0 Lost | Select All Filters | 5 Items

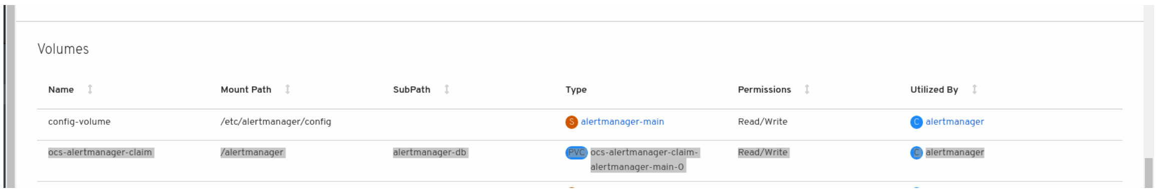
Name	Namespace	Status	Persistent Volume	Requested
PVC my-alertmanager-claim-alertmanager-main-0	NS openshift-monitoring	Bound	PV pvc-d00428a5-0ce6-11ea-8fe8-023bdfa29edc	40Gi
PVC my-alertmanager-claim-alertmanager-main-1	NS openshift-monitoring	Bound	PV pvc-d00be111-0ce6-11ea-8fe8-023bdfa29edc	40Gi
PVC my-alertmanager-claim-alertmanager-main-2	NS openshift-monitoring	Bound	PV pvc-d01ac717-0ce6-11ea-8fe8-023bdfa29edc	40Gi
PVC my-prometheus-claim-prometheus-k8s-0	NS openshift-monitoring	Bound	PV pvc-ce290f1b-0ce6-11ea-8fe8-023bdfa29edc	40Gi
PVC my-prometheus-claim-prometheus-k8s-1	NS openshift-monitoring	Bound	PV pvc-ce361010-0ce6-11ea-8fe8-023bdfa29edc	40Gi

2. Verify that the new **alertmanager-main-*** pods appear with a state of **Running**.

   a. Click the new **alertmanager-main-*** pods to view the pod details.

   b. Scroll down to **Volumes** and verify that the volume has a **Type**, **ocs-alertmanager-claim** that matches one of your new Persistent Volume Claims, for example, **ocs-alertmanager-claim-alertmanager-main-0**.

   Persistent Volume Claims attached to alertmanager-main-* pod


Volumes

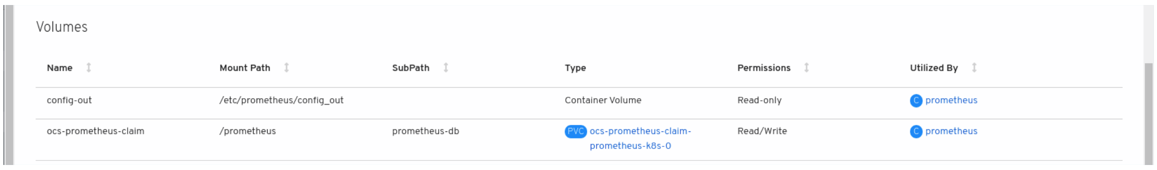
Name	Mount Path	SubPath	Type	Permissions	Utilized By
config-volume	/etc/alertmanager/config		S alertmanager-main	Read/Write	C alertmanager
ocs-alertmanager-claim	/alertmanager	alertmanager-db	PVC ocs-alertmanager-claim-alertmanager-main-0	Read/Write	C alertmanager

3. Verify that the new **prometheus-k8s-*** pods appear with a state of **Running**.

   a. Click the new **prometheus-k8s-*** pods to view the pod details.

   b. Scroll down to **Volumes** and verify that the volume has a **Type**, **ocs-prometheus-claim** that matches one of your new Persistent Volume Claims, for example, **ocs-prometheus-claim-prometheus-k8s-0**.

   Persistent Volume Claims attached to prometheus-k8s-* pod


Volumes

Name	Mount Path	SubPath	Type	Permissions	Utilized By
config-out	/etc/prometheus/config_out		Container Volume	Read-only	C prometheus
ocs-prometheus-claim	/prometheus	prometheus-db	PVC ocs-prometheus-claim-prometheus-k8s-0	Read/Write	C prometheus

2.3. CLUSTER LOGGING FOR OPENSHIFT CONTAINER STORAGE

You can deploy cluster logging to aggregate logs for a range of OpenShift Container Platform services. For information about how to deploy cluster logging, see Deploying cluster logging.

Upon initial OpenShift Container Platform deployment, OpenShift Container Storage is not configured by default and the OpenShift Container Platform cluster will solely rely on default storage available from the nodes. You can edit the default configuration of OpenShift logging (ElasticSearch) to be backed by OpenShift Container Storage to have OpenShift Container Storage backed logging (Elasticsearch).

IMPORTANT

Always ensure that you have plenty of storage capacity for these services. If you run out of storage space for these critical services, the logging application becomes inoperable and very difficult to recover.

Red Hat recommends configuring shorter curation and retention intervals for these services. See Configuring Curator in the OpenShift Container Platform documentation for details.

If you run out of storage space for these services, contact Red Hat Customer Support.

2.3.1. Configuring persistent storage

You can configure a persistent storage class and size for the Elasticsearch cluster using the storage class name and size parameters. The Cluster Logging Operator creates a Persistent Volume Claim for each data node in the Elasticsearch cluster based on these parameters. For example:

```
spec:
    logStore:
      type: "elasticsearch"
      elasticsearch:
        nodeCount: 3
        storage:
          storageClassName: "ocs-storagecluster-ceph-rbd”
          size: "200G"
```

This example specifies that each data node in the cluster will be bound to a Persistent Volume Claim that requests **200GiB** of **ocs-storagecluster-ceph-rbd** storage. Each primary shard will be backed by a single replica. A copy of the shard is replicated across all the nodes and are always available and the copy can be recovered if at least two nodes exist due to the single redundancy policy. For information about Elasticsearch replication policies, see Elasticsearch replication policy in About deploying and configuring cluster logging.

NOTE

Omission of the storage block will result in a deployment backed by default storage. For example:

```
spec:
    logStore:
      type: "elasticsearch"
      elasticsearch:
        nodeCount: 3
        storage: {}
```

For more information, see Configuring cluster logging.

2.3.2. Configuring cluster logging to use OpenShift Container Storage

Follow the instructions in this section to configure OpenShift Container Storage as storage for the OpenShift cluster logging.

NOTE

You can obtain all the logs when you configure logging for the first time in OpenShift Container Storage. However, after you uninstall and reinstall logging, the old logs are removed and only the new logs are processed.

Prerequisites

- You have administrative access to OpenShift Web Console.
- OpenShift Container Storage Operator is installed and running in the **openshift-storage** namespace.
- Cluster logging Operator is installed and running in the **openshift-logging** namespace.

Procedure

1. Click **Administration → Custom Resource Definitions**from the left pane of the OpenShift Web Console.
2. On the Custom Resource Definitions page, click **ClusterLogging**.
3. On the Custom Resource Definition Overview page, select **View Instances** from the Actions menu or click the **Instances** Tab.
4. On the Cluster Logging page, click **Create Cluster Logging**.
   You might have to refresh the page to load the data.
5. In the YAML, replace the code with the following:

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      storage:
        storageClassName: ocs-storagecluster-ceph-rbd
        size: 200G
      redundancyPolicy: "SingleRedundancy"
  visualization:
    type: "kibana"
    kibana:
      replicas: 1
  curation:
    type: "curator"
    curator:
      schedule: "30 3 * * *"
  collection:
    logs:
      type: "fluentd"
      fluentd: {}
```

6. Click **Save**.

Verification steps

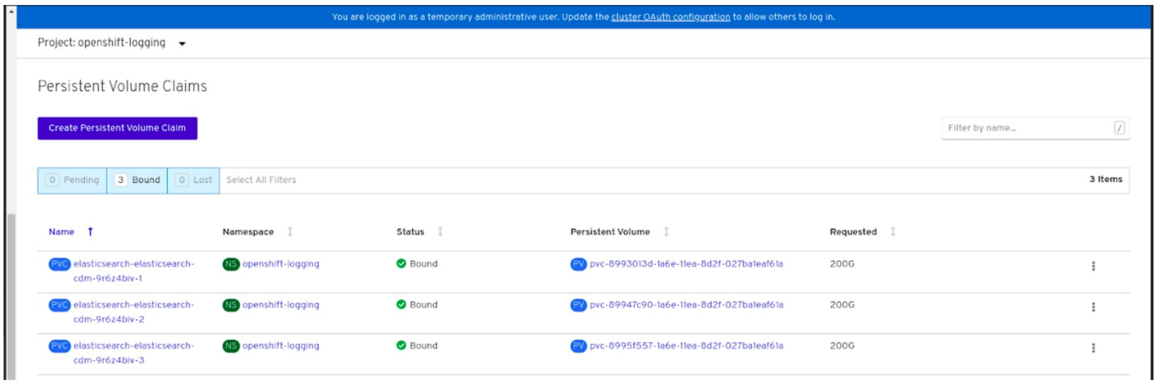
1. Verify that the Persistent Volume Claims are bound to the **elasticsearch** pods.

   a. Go to **Storage → Persistent Volume Claims**.

   b. Set the **Project** dropdown to **openshift-logging**.

   c. Verify that Persistent Volume Claims are visible with a state of **Bound**, attached to **elasticsearch-*** pods.

   Figure 2.1. Cluster logging created and bound

   You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

   Project: openshift-logging

   Persistent Volume Claims

   Create Persistent Volume Claim

   Filter by name...

   0 Pending | 3 Bound | 0 Lost | Select All Filters — 3 Items

Name	Namespace	Status	Persistent Volume	Requested
PVC elasticsearch-elasticsearch-cdm-9r6z4biv-1	NS openshift-logging	Bound	PV pvc-8993013d-1a6e-11ea-8d2f-027ba1eaf61a	200G
PVC elasticsearch-elasticsearch-cdm-9r6z4biv-2	NS openshift-logging	Bound	PV pvc-89947c90-1a6e-11ea-8d2f-027ba1eaf61a	200G
PVC elasticsearch-elasticsearch-cdm-9r6z4biv-3	NS openshift-logging	Bound	PV pvc-8995f557-1a6e-11ea-8d2f-027ba1eaf61a	200G

2. Verify that the new cluster logging is being used.

   a. Click **Workload → Pods**.

   b. Set the Project to **openshift-logging**.

   c. Verify that the new **elasticsearch-*** pods appear with a state of **Running**.

   d. Click the new **elasticsearch-*** pod to view pod details.

   e. Scroll down to **Volumes** and verify that the elasticsearch volume has a **Type** that matches your new Persistent Volume Claim, for example, **elasticsearch-elasticsearch-cdm-9r624biv-3**.

   f. Click the Persistent Volume Claim name and verify the storage class name in the PersistenVolumeClaim Overview page.

NOTE

Make sure to use a shorter curator time to avoid PV full scenario on PVs attached to Elasticsearch pods.

You can configure Curator to delete Elasticsearch data based on retention settings. It is recommended that you set the following default index data retention of 5 days as a default.

```
config.yaml: |
    openshift-storage:
      delete:
        days: 5
```

For more details, see Curation of Elasticsearch Data .

NOTE

To uninstall cluster logging backed by Persistent Volume Claim, use the steps in Removing the cluster logging operator from OpenShift Container Storage .

2.4. OBJECT BUCKET CLAIM

OpenShift Container Storage introduces a new concept called Object Bucket Claim. An Object Bucket Claim can be used to request an S3 compatible bucket backend for your workloads.

You can create an Object Bucket Claim two ways:

- Section 2.4.1, “Dynamic Object Bucket Claim”
- Section 2.4.2, “Creating an Object Bucket Claim using the command line interface”

2.4.1. Dynamic Object Bucket Claim

Similar to persistent volumes, you can add the details of the Object Bucket claim to your application’s YAML, and get the object service endpoint, access key, and secret access key available in a configuration map and secret. It is easy to read this information dynamically into environment variables of your application.

Procedure

1. Add the following lines to your application YAML:

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: <obc-name>
spec:
  generateBucketName: <obc-bucket-name>
  storageClassName: noobaa
```

These lines are the Object Bucket Claim itself.

a. Replace **<obc-name>** with the a unique Object Bucket Claim name.

   b. Replace **<obc-bucket-name>** with a unique bucket name for your Object Bucket Claim.

2. You can add more lines to the YAML file to automate the use of the Object Bucket Claim. The example below is the mapping between the bucket claim result, which is a configuration map with data and a secret with the credentials. This specific job will claim the Object Bucket from NooBaa, which will create a bucket and an account.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: testjob
spec:
  template:
    spec:
      restartPolicy: OnFailure
      containers:
        - image: <your application image>
          name: test
          env:
            - name: BUCKET_NAME
              valueFrom:
                configMapKeyRef:
                  name: <obc-name>
                  key: BUCKET_NAME
            - name: BUCKET_HOST
              valueFrom:
                configMapKeyRef:
                  name: <obc-name>
                  key: BUCKET_HOST
            - name: BUCKET_PORT
              valueFrom:
                configMapKeyRef:
                  name: <obc-name>
                  key: BUCKET_PORT
            - name: AWS_ACCESS_KEY_ID
              valueFrom:
                secretKeyRef:
                  name: <obc-name>
                  key: AWS_ACCESS_KEY_ID
            - name: AWS_SECRET_ACCESS_KEY
              valueFrom:
                secretKeyRef:
                  name: <obc-name>
                  key: AWS_SECRET_ACCESS_KEY
```

   a. Replace all instances of <obc-name> with your Object Bucket Claim name.

   b. Replace <your application image> with your application image.

3. Apply the updated YAML file:

```
# oc apply -f <yaml.file>
```

   a. Replace **<yaml.file>** with the name of your YAML file.

4. To view the new configuration map, run the following:

```
# oc get cm <obc-name>
```

a. Replace **obc-name** with the name of your Object Bucket Claim.
   You can expect the following environment variables in the output:

   - **BUCKET_HOST** - Endpoint to use in the application
   - **BUCKET_PORT** - The port available for the application
     - The port is related to the **BUCKET_HOST**. For example, if the **BUCKET_HOST** is https://my.example.com, and the **BUCKET_PORT** is 443, the endpoint for the object service would be https://my.example.com:443.
   - **BUCKET_NAME** - Requested or generated bucket name
   - **AWS_ACCESS_KEY_ID** - Access key that is part of the credentials
   - **AWS_SECRET_ACCESS_KEY** - Secret access key that is part of the credentials

2.4.2. Creating an Object Bucket Claim using the command line interface

When creating an Object Bucket Claim using the command-line interface, you get a configuration map and a Secret that together contain all the information your application needs to use the object storage service.

Prerequisites

- Download the MCG command-line interface:

```
# subscription-manager repos --enable=rh-ocs-4-for-rhel-8-x86_64-rpms
# yum install mcg
```

Procedure

1. Use the command-line interface to generate the details of a new bucket and credentials. Run the following command:

   ```
   # noobaa obc create <obc-name> -n openshift-storage
   ```

   Replace **<obc-name>** with a unique Object Bucket Claim name, for example, **myappobc**.

   Additionally, you can use the **--app-namespace** option to specify the namespace where the Object Bucket Claim configuration map and secret will be created, for example, **myapp-namespace**.

   Example output:

   ```
   INFO[0001]  Created: ObjectBucketClaim "test21obc"
   ```

   The MCG command-line-interface has created the necessary configuration and has informed OpenShift about the new OBC.

2. Run the following command to view the Object Bucket Claim:

```
# oc get obc -n openshift-storage
```

Example output:

```
NAME       STORAGE-CLASS                 PHASE   AGE
test21obc   openshift-storage.noobaa.io   Bound   38s
```

3. Run the following command to view the YAML file for the new Object Bucket Claim:

```
# oc get obc test21obc -o yaml -n openshift-storage
```

Example output:

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  creationTimestamp: "2019-10-24T13:30:07Z"
  finalizers:
  - objectbucket.io/finalizer
  generation: 2
  labels:
    app: noobaa
    bucket-provisioner: openshift-storage.noobaa.io-obc
    noobaa-domain: openshift-storage.noobaa.io
  name: test21obc
  namespace: openshift-storage
  resourceVersion: "40756"
  selfLink: /apis/objectbucket.io/v1alpha1/namespaces/openshift-
storage/objectbucketclaims/test21obc
  uid: 64f04cba-f662-11e9-bc3c-0295250841af
spec:
  ObjectBucketName: obc-openshift-storage-test21obc
  bucketName: test21obc-933348a6-e267-4f82-82f1-e59bf4fe3bb4
  generateBucketName: test21obc
  storageClassName: openshift-storage.noobaa.io
status:
  phase: Bound
```

4. Inside of your **openshift-storage** namespace, you can find the configuration map and the secret to use this Object Bucket Claim. The CM and the secret have the same name as the Object Bucket Claim. To view the secret:

```
# oc get -n openshift-storage secret test21obc -o yaml
```

Example output:

```
Example output:
apiVersion: v1
data:
  AWS_ACCESS_KEY_ID: c0M0R2xVanF3ODR3bHBkVW94cmY=
  AWS_SECRET_ACCESS_KEY:
Wi9kcFluSWxHRzlWaFlzNk1hc0xma2JXcjM1MVhqa051SlBleXpmOQ==
kind: Secret
metadata:
  creationTimestamp: "2019-10-24T13:30:07Z"
  finalizers:
  - objectbucket.io/finalizer
  labels:
    app: noobaa
    bucket-provisioner: openshift-storage.noobaa.io-obc
    noobaa-domain: openshift-storage.noobaa.io
  name: test21obc
  namespace: openshift-storage
  ownerReferences:
  - apiVersion: objectbucket.io/v1alpha1
    blockOwnerDeletion: true
    controller: true
    kind: ObjectBucketClaim
    name: test21obc
    uid: 64f04cba-f662-11e9-bc3c-0295250841af
  resourceVersion: "40751"
  selfLink: /api/v1/namespaces/openshift-storage/secrets/test21obc
  uid: 65117c1c-f662-11e9-9094-0a5305de57bb
type: Opaque
```

The secret gives you the S3 access credentials.

5. To view the configuration map:

```
# oc get -n openshift-storage cm test21obc -o yaml
```

Example output:

```
apiVersion: v1
data:
  BUCKET_HOST: 10.0.171.35
  BUCKET_NAME: test21obc-933348a6-e267-4f82-82f1-e59bf4fe3bb4
  BUCKET_PORT: "31242"
  BUCKET_REGION: ""
  BUCKET_SUBREGION: ""
kind: ConfigMap
metadata:
  creationTimestamp: "2019-10-24T13:30:07Z"
  finalizers:
  - objectbucket.io/finalizer
  labels:
    app: noobaa
    bucket-provisioner: openshift-storage.noobaa.io-obc
    noobaa-domain: openshift-storage.noobaa.io
  name: test21obc
  namespace: openshift-storage
  ownerReferences:
  - apiVersion: objectbucket.io/v1alpha1
    blockOwnerDeletion: true
    controller: true
    kind: ObjectBucketClaim
    name: test21obc
    uid: 64f04cba-f662-11e9-bc3c-0295250841af
```

```
resourceVersion: "40752"
selfLink: /api/v1/namespaces/openshift-storage/configmaps/test21obc
uid: 651c6501-f662-11e9-9094-0a5305de57bb
```

The configuration map contains the S3 endpoint information for your application.

CHAPTER 3. BACKING OPENSHIFT CONTAINER PLATFORM APPLICATIONS WITH OPENSHIFT CONTAINER STORAGE

You cannot directly install OpenShift Container Storage during the OpenShift Container Platform installation. However, you can install OpenShift Container Storage on an existing OpenShift Container Platform by using the Operator Hub and then configure the OpenShift Container Platform applications to be backed by OpenShift Container Storage.

Prerequisites

- OpenShift Container Platform is installed and you have administrative access to OpenShift Web Console.
- OpenShift Container Storage is installed and running in the **openshift-storage** namespace.

Procedure

1. In the OpenShift Web Console, perform one of the following:
   - Click **Workloads → Deployments**.
     In the Deployments page, you can do one of the following:
     - Select any existing deployment and click **Add Storage** option from the **Action** menu (⋮).
     - Create a new deployment and then add storage.
       i. Click **Create Deployment** to create a new deployment.
       ii. Edit the **YAML** based on your requirement to create a deployment.
       iii. Click **Create**.
       iv. Select **Add Storage** from the **Actions** drop down menu on the top right of the page.
   - Click **Workloads → Deployment Configs**
     In the Deployment Configs page, you can do one of the following:
     - Select any existing deployment and click **Add Storage** option from the **Action** menu (⋮).
     - Create a new deployment and then add storage.
       i. Click **Create Deployment Config** to create a new deployment.
       ii. Edit the **YAML** based on your requirement to create a deployment.
       iii. Click **Create**.
       iv. Select **Add Storage** from the **Actions** drop down menu on the top right of the page.
2. In the Add Storage page, you can choose one of the following options:
   - Click the **Use existing claim** option and select a suitable PVC from the drop down list.

- Click the **Create new claim** option.
    a. Select **ocs-storagecluster-ceph-rbd** or **ocs-storagecluster-cephfs** storage class from the **Storage Class** drop down list.
    b. Provide a name for the Persistent Volume Claim.
    c. Select ReadWriteOnce (RWO) or ReadWriteMany (RWX) access mode.

    NOTE

    ReadOnlyMany (ROX) is deactivated as it is not supported.

    d. Select the size of the desired storage capacity.

    NOTE

    You cannot resize the storage capacity after the creation of Persistent Volume Claim.

3. Specify the mount path and subpath (if required) for the mount path volume inside the container.
4. Click **Save**.

Verification steps

1. Depending on your configuration, perform one of the following:
    - Click **Workloads → Deployments**.
    - Click **Workloads → Deployment Configs**
2. Set the Project as required.
3. Click the deployment for you which you added storage to view the deployment details.
4. Scroll down to **Volumes** and verify that your deployment has a **Type** that matches the Persistent Volume Claim that you assigned.
5. Click the Persistent Volume Claim name and verify the storage class name in the PersistenVolumeClaim Overview page.

CHAPTER 4. SCALING STORAGE NODES

To scale the storage capacity of OpenShift Container Storage, you can do either of the following:

- **Scale up storage nodes** - Add storage capacity to the existing Red Hat OpenShift Container Storage worker nodes
- **Scale out storage nodes** - Add new worker nodes to increase the storage capacity

4.1. REQUIREMENTS FOR SCALING STORAGE NODES

Before you proceed to scale the storage nodes, refer to the following sections to understand the node requirements for your specific Red Hat OpenShift Container Storage instance:

- Supported configurations
- Infrastructure requirements
- Sizing and scaling recommendations
- Software requirements

WARNING

Always ensure that you have plenty of storage capacity.

If storage ever fills completely, it is not possible to add capacity or delete or migrate content away from the storage to free up space. Completely full storage is very difficult to recover.

Capacity alerts are issued when cluster storage capacity reaches 75% (near-full) and 85% (full) of total capacity. Always address capacity warnings promptly, and review your storage regularly to ensure that you do not run out of storage space.

If you do run out of storage space completely, contact Red Hat Customer Support.

4.1.1. Supported Deployments for Red Hat OpenShift Container Storage

- User-provisioned infrastructure:
  - Amazon Web Services (AWS)
  - VMware
- Installer-provisioned infrastructure:
  - Amazon Web Services (AWS)

4.2. SCALING UP STORAGE BY ADDING CAPACITY TO YOUR OPENSHIFT CONTAINER STORAGE NODES

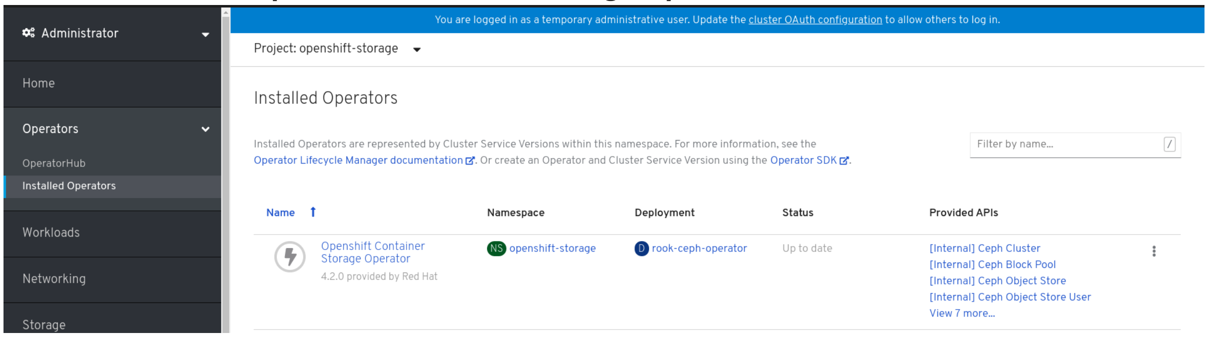
Use this procedure to add storage capacity and performance to your configured Red Hat OpenShift Container Storage worker nodes.

Prerequisites

- A running OpenShift Container Storage Platform
- Administrative privileges on the OpenShift Web Console

Procedure

1. Navigate to the OpenShift Web Console.
2. Click on **Operators** on the left navigation bar.
3. Select **Installed Operators**.
4. In the window, click **OpenShift Container Storage Operator**:

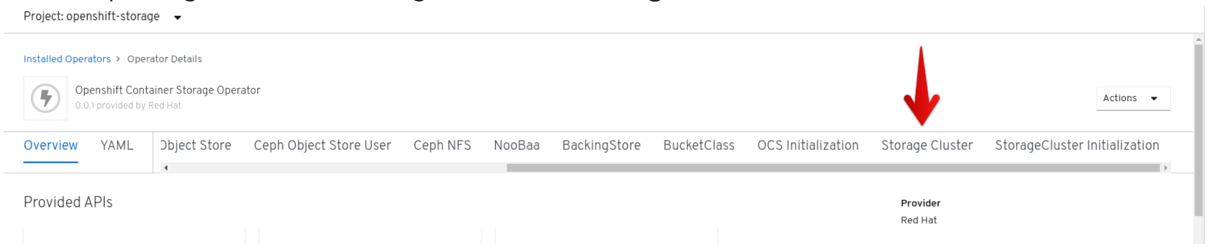


You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Installed Operators are represented by Cluster Service Versions within this namespace. For more information, see the Operator Lifecycle Manager documentation. Or create an Operator and Cluster Service Version using the Operator SDK.

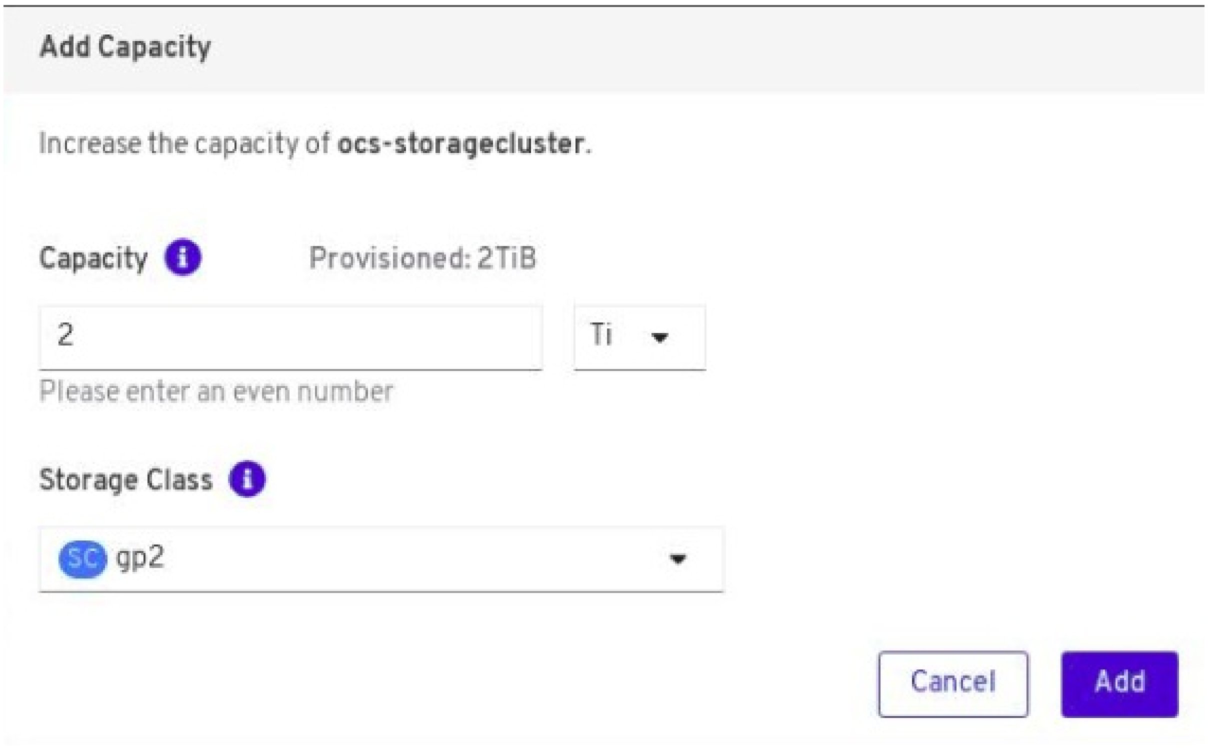
Name	Namespace	Deployment	Status	Provided APIs
Openshift Container Storage Operator 4.2.0 provided by Red Hat	NS openshift-storage	D rook-ceph-operator	Up to date	[Internal] Ceph Cluster [Internal] Ceph Block Pool [Internal] Ceph Object Store [Internal] Ceph Object Store User View 7 more...

5. In the top navigation bar, scroll right and click **Storage Cluster** tab.



Project: openshift-storage
Installed Operators > Operator Details
Openshift Container Storage Operator
0.0.1 provided by Red Hat
Actions
Overview
YAML
Object Store
Ceph Object Store User
Ceph NFS
NooBaa
BackingStore
BucketClass
OCS Initialization
Storage Cluster
StorageCluster Initialization
Provided APIs
Provider
Red Hat

6. The visible list should have only one item. Click (⋮) on the far right to extend the options menu.
7. Select **Add Capacity** from the options menu.


Add Capacity

Increase the capacity of **ocs-storagecluster**.

Capacity: 2 Ti
Provisioned: 2TiB
Please enter an even number

Storage Class: SC gp2

From this dialog box, you can set the requested additional capacity and the storage class. The size should always be set in multiples of 2TiB. On AWS, the storage class should be set to **gp2**. On VMWare, the storage class should be set to **thin**.

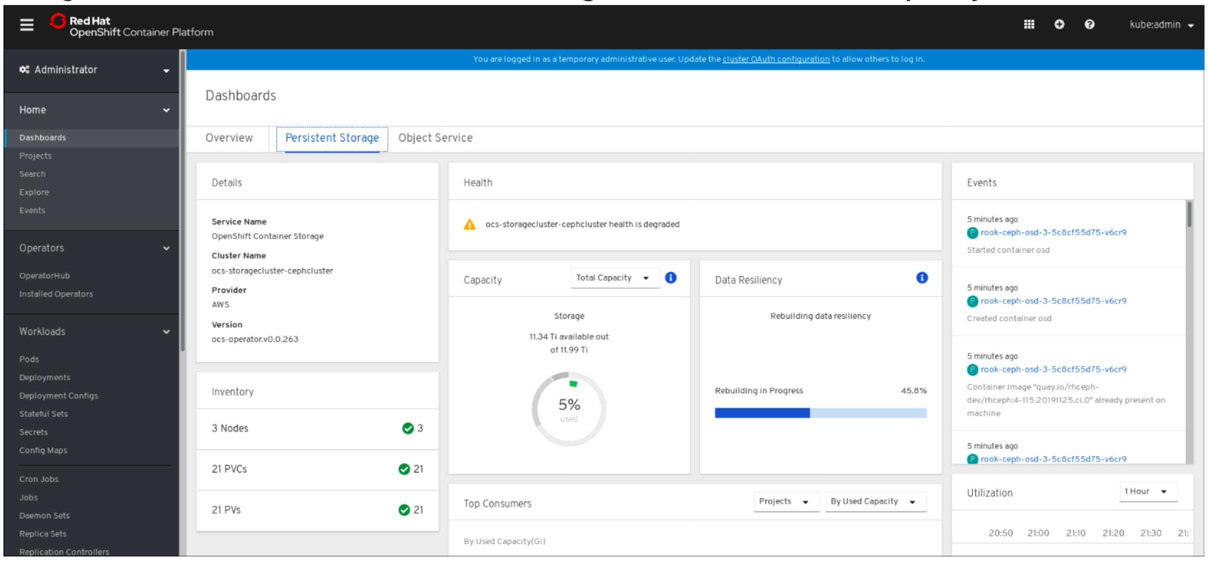
NOTE

The effectively provisioned capacity will be three times as much as you put into the **Requested Capacity** field because OpenShift Container Storage uses a replica count of 3.

8. Once you are done with your setting, click **Add**. You will not see the status of the storage cluster until it reaches **Ready**. You might need to wait a couple of minutes after you see the **Ready** state.

Verification steps

1. Navigate to **Dashboards** -→ **Persistent Storage** tab, then check the **Capacity** card.



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Dashboards

Details
Service Name: OpenShift Container Storage
Cluster Name: ocs-storagecluster-cephcluster
Provider: AWS
Version: ocs-operator.v0.0.263

Inventory
3 Nodes ✓ 3
21 PVCs ✓ 21
21 PVs ✓ 21

Health
ocs-storagecluster-cephcluster health is degraded

Capacity
Storage
11.34 Ti available out of 11.99 Ti
5% used

Data Resiliency
Rebuilding data resiliency
Rebuilding in Progress 45.8%

Top Consumers
By Used Capacity(Gi)

Events
5 minutes ago
rook-ceph-osd-3-5c8cf55d75-v6cr9
Started container osd

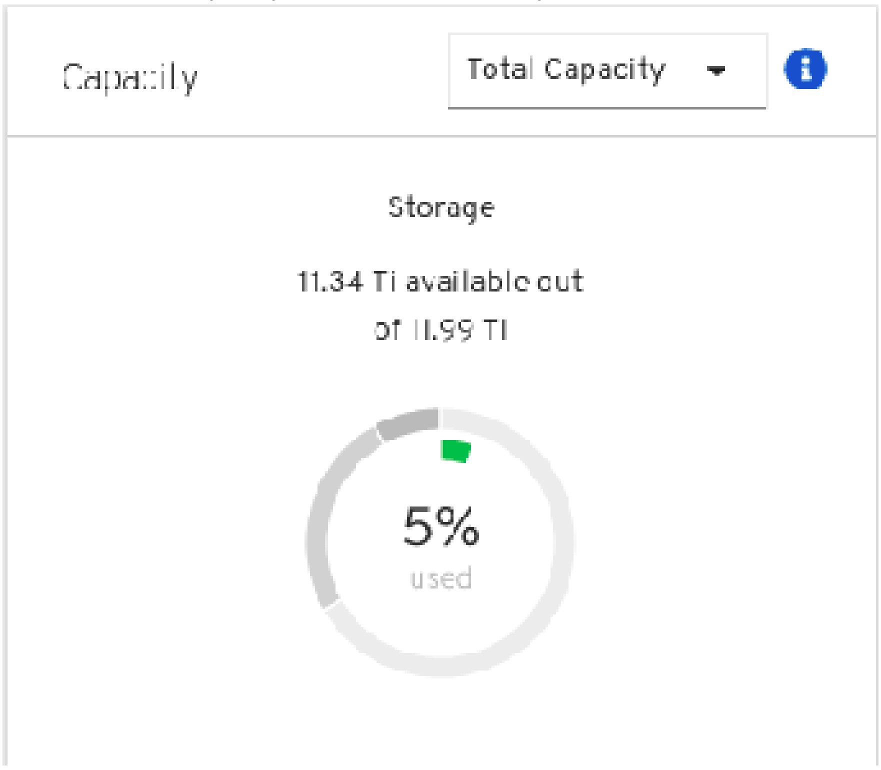
5 minutes ago
rook-ceph-osd-3-5c8cf55d75-v6cr9
Created container osd

5 minutes ago
rook-ceph-osd-3-5c8cf55d75-v6cr9
Container image "quay.io/rhceph-dev/rhceph:4-115.20191125.ci.0" already present on machine

5 minutes ago
rook-ceph-osd-3-5c8cf55d75-v6cr9

Utilization
20:50 21:00 21:10 21:20 21:30 21:

2. Note that the capacity increases based on your selections.



IMPORTANT

As of OpenShift Container Storage 4.2, cluster reduction, whether by reducing OSDs or nodes, is not supported.

4.3. SCALING OUT STORAGE CAPACITY

To scale out storage capacity, you need to perform the following steps:

- Add a new node
- Verify that the new node is added successfully
- Scale up the storage capacity

4.3.1. Adding a node

You can add additional nodes to increase the storage capacity when existing worker nodes are already running at their maximum supported OSDs, which is 3 OSDs of size 2TiB.

Depending on the type of your deployment, you can choose one of the following procedures to add a storage node:

- For AWS installer-provisioned infrastructure, see Section 4.3.1.1, “Adding a node on an AWS installer-provisioned infrastructure”
- For AWS or VMware user-provisioned infrastructure, see Section 4.3.1.2, “Adding a node on an AWS or a VMware user-provisioned infrastructure”

4.3.1.1. Adding a node on an AWS installer-provisioned infrastructure

Prerequisites

- A running OpenShift Container Storage Platform
- Access to the OpenShift Web Console

Procedure

1. Navigate to the OpenShift Web Console.
2. Navigate to **Compute** → **Machine Sets**.
3. On the machine set where you want to add nodes, click the Action menu **(⋮)** → **Edit Count** beside the required machine.
4. Add the amount of nodes, and click **Save**.
5. Click **Compute** → **Nodes** and confirm if the new node is in **Ready** state.

   NOTE

   It might take some time for the new node to reach the **Ready** state.

6. Apply the OpenShift Container Storage label to the new node using any one of the following:

   From User interface

   a. For the new node, click **Action Menu (⋮)** → **Edit Labels**

   b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

   From Command line interface

   - Execute the following command to apply the OpenShift Container Storage label to the new node:

     ```
     $ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
     ```

     NOTE

     It is recommended to add 3 worker nodes. For AWS deployments with three availability zones, make sure that you add each node to a different availability zone and perform this procedure on all the nodes. However, for AWS deployments with one availability zone, you add the 3 worker nodes to the available zone and perform this procedure on that zone.

Verification steps

To verify that the new node is added, see Section 4.3.2, “Verifying the addition of a new node” .

4.3.1.2. Adding a node on an AWS or a VMware user-provisioned infrastructure

Prerequisites

- A running OpenShift Container Storage Platform
- Access to the OpenShift Web Console

Procedure

1. Depending on whether you are adding a node on an AWS user provisioned infrastructure or a VMware user-provisioned infrastructure, perform the following steps:
   - For AWS
     a. Create a new AWS machine instance with the required infrastructure. See Infrastructure requirements.
     b. Create a new OpenShift Container Platform node using the new AWS machine instance.
   - For VMware:
     a. Create a new VM on vSphere with the required infrastructure. See Infrastructure requirements.
     b. Create a new OpenShift Container Platform worker node using the new VM.
2. Check for certificate signing requests (CSRs) that are in **Pending** state:
   ```
   $ oc get csr
   ```
3. Approve all required CSRs for the new node:
   ```
   $ oc adm certificate approve <Certificate_Name>
   ```
4. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.
5. Apply the OpenShift Container Storage label to the new node using any one of the following:

   From User interface

   a. For the new node, click **Action Menu (⋮) → Edit Labels**
   b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

   From Command line interface

   - Execute the following command to apply the OpenShift Container Storage label to the new node:
     ```
     $ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
     ```

     NOTE

     It is recommended to add 3 worker nodes. For AWS deployments with three availability zones, make sure that you add each node to a different availability zone and perform this procedure on all the nodes. However, for AWS deployments with one availability zone and VMware deployments that have only one availability zone, you add the 3 worker nodes to the available zone and perform this procedure on that zone.

Verification steps

To verify that the new node is added, see Section 4.3.2, "Verifying the addition of a new node" .

4.3.2. Verifying the addition of a new node

1. Execute the following command and verify that the new node is present in the output:

```
$ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***
3. Verify the health of OpenShift Container Storage cluster using the persistent storage dashboard:
   a. Click **Home → Dashboards** from the left pane of the OpenShift Web Console and click the **Persistent Storage** tab.
   b. In the **Health card**, verify that the cluster health is displayed as *ocs-storagecluster is healthy*.

4.3.3. Scaling up storage capacity

To scale up storage capacity, perform the steps in the procedure, Scaling up storage by adding capacity to your OpenShift Container Storage nodes.

CHAPTER 5. MANAGING PERSISTENT VOLUME CLAIMS

IMPORTANT

Expanding PVCs is not supported for PVCs backed by OpenShift Container Storage 4.2

5.1. CONFIGURING APPLICATION PODS TO USE OPENSHIFT CONTAINER STORAGE

Follow the instructions in this section to configure OpenShift Container Storage as storage for an application pod.

Prerequisites

- You have administrative access to OpenShift Web Console.
- OpenShift Container Storage Operator is installed and running in the **openshift-storage** namespace. In OpenShift Web Console, click **Operators → Installed Operators** to view installed operators.
- The default storage classes provided by OpenShift Container Storage are available. In OpenShift Web Console, click **Storage → Storage Classes** to view default storage classes.

Procedure

1. **Create a Persistent Volume Claim (PVC) for the application to use.**
    a. In OpenShift Web Console, click **Storage → Persistent Volume Claims**.
    b. Set the **Project** for the application pod.
    c. Click **Create Persistent Volume Claim**
        i. Specify a **Storage Class** provided by OpenShift Container Storage.
        ii. Specify the PVC **Name**, for example, **myclaim**.
        iii. Select the required **Access Mode**.
        iv. Specify a **Size** as per application requirement.
        v. Click **Create** and wait until the PVC is in **Bound** status.
2. **Configure a new or existing application pod to use the new PVC.**
    - For a new application pod, perform the following steps:
        i. Click **Workloads →Pods**.
        ii. Create a new application pod.
        iii. Under the **spec:** section, add **volume:** section to add the new PVC as a volume for the application pod.

```
volumes:
  - name: <volume_name>
    persistentVolumeClaim:
      claimName: <pvc_name>
```

For example:

```
volumes:
  - name: mypd
    persistentVolumeClaim:
      claimName: myclaim
```

- For an existing application pod, perform the following steps:

  i. Click **Workloads →Deployment Configs**.

  ii. Search for the required deployment config associated with the application pod.

  iii. Click on its **Action menu (⋮) → Edit Deployment Config**.

  iv. Under the **spec:** section, add **volume:** section to add the new PVC as a volume for the application pod and click **Save**.

```
volumes:
  - name: <volume_name>
    persistentVolumeClaim:
      claimName: <pvc_name>
```

For example:

```
volumes:
  - name: mypd
    persistentVolumeClaim:
      claimName: myclaim
```

3. **Verify that the new configuration is being used.**

   a. Click **Workloads → Pods**.

   b. Set the **Project** for the application pod.

   c. Verify that the application pod appears with a status of **Running**.

   d. Click the application pod name to view pod details.

   e. Scroll down to **Volumes** section and verify that the volume has a **Type** that matches your new Persistent Volume Claim, for example, **myclaim**.

5.2. VIEWING PERSISTENT VOLUME CLAIM REQUEST STATUS

WARNING

Expanding Persistent Volume Claims (PVCs) is not supported for PVCs backed by OpenShift Container Storage 4.2

Use this procedure to view the status of a PVC request.

Prerequisites

- Administrator access to OpenShift Container Storage.

Procedure

1. Log in to OpenShift Web Console.
2. Click **Storage → Persistent Volume Claims**
3. Search for the required PVC name by using the **Filter** textbox.
4. Check the **Status** column corresponding to the required PVC.
5. Click the required **Name** to view the PVC details.

5.3. REVIEWING PERSISTENT VOLUME CLAIM REQUEST EVENTS

Use this procedure to review and address Persistent Volume Claim (PVC) request events.

Prerequisites

- Administrator access to OpenShift Web Console.

Procedure

1. Log in to OpenShift Web Console.
2. Click **Home → Dashboards → Persistent Storage**
3. Locate the **Inventory** card to see the number of PVCs with errors.
4. Click **Storage → Persistent Volume Claims**
5. Search for the required PVC using the **Filter** textbox.
6. Click on the PVC name and navigate to **Events**
7. Address the events as required or as directed.

5.4. DYNAMIC PROVISIONING

5.4.1. About dynamic provisioning

The StorageClass resource object describes and classifies storage that can be requested, as well as provides a means for passing parameters for dynamically provisioned storage on demand. StorageClass objects can also serve as a management mechanism for controlling different levels of storage and access to the storage. Cluster Administrators (**cluster-admin**) or Storage Administrators (**storage-admin**) define and create the StorageClass objects that users can request without needing any intimate knowledge about the underlying storage volume sources.

The OpenShift Container Platform persistent volume framework enables this functionality and allows administrators to provision a cluster with persistent storage. The framework also gives users a way to request those resources without having any knowledge of the underlying infrastructure.

Many storage types are available for use as persistent volumes in OpenShift Container Platform. While all of them can be statically provisioned by an administrator, some types of storage are created dynamically using the built-in provider and plug-in APIs.

5.4.2. Dynamic provisioning in OpenShift Container Storage

Red Hat OpenShift Container Storage is software-defined storage that is optimised for container environments. It runs as an operator on OpenShift Container Platform to provide highly integrated and simplified persistent storage management for containers.

OpenShift Container Storage supports a variety of storage types, including:

- Block storage for databases
- Shared file storage for continuous integration, messaging, and data aggregation
- Object storage for archival, backup, and media storage

Version 4.2 uses Red Hat Ceph Storage to provide the file, block, and object storage that backs persistent volumes, and Rook.io to manage and orchestrate provisioning of persistent volumes and claims. NooBaa provides object storage in the Multicloud Object Gateway.

In OpenShift Container Storage 4.2, the Red Hat Ceph Storage Container Storage Interface (CSI) driver for RADOS Block Device (RBD) and Ceph File System (CephFS) handles the dynamic provisioning requests. When a PVC request comes in dynamically, the CSI driver has the following options:

- Create a PVC with ReadWriteOnce (RWO) and ReadWriteMany (RWX) access that is based on Ceph RBDs with volume mode **Block**
- Create a PVC with ReadWriteOnce (RWO) access that is based on Ceph RBDs with volume mode **Filesystem**
- Create a PVC with ReadWriteOnce (RWO) and ReadWriteMany (RWX) access that is based on CephFS for volume mode **Filesystem**

The judgement of which driver (RBD or CephFS) to use is based on the entry in the **storageclass.yaml** file.

5.4.3. Available dynamic provisioning plug-ins

OpenShift Container Platform provides the following provisioner plug-ins, which have generic implementations for dynamic provisioning that use the cluster’s configured provider’s API to create new storage resources:

Storage type	Provisioner plug-in name	Notes
AWS Elastic Block Store (EBS)	**kubernetes.io/aws-ebs**	For dynamic provisioning when using multiple clusters in different zones, tag each node with **Key=kubernetes.io/cluster/<cluster_name>,Value=<cluster_id>** where **<cluster_name>** and **<cluster_id>** are unique per cluster.
AWS Elastic File System (EFS)		Dynamic provisioning is accomplished through the EFS provisioner pod and not through a provisioner plug-in.
Azure Disk	**kubernetes.io/azure-disk**	
Azure File	**kubernetes.io/azure-file**	The **persistent-volume-binder** ServiceAccount requires permissions to create and get Secrets to store the Azure storage account and keys.
Ceph File System (POSIX Compliant filesystem)	**openshift-storage.cephfs.csi.ceph.com**	Provisions a volume for ReadWriteMany (RWX) or ReadWriteOnce (RWO) access modes using the Ceph Filesytem configured in a Ceph cluster.
Ceph RBD (Block Device)	**openshift-storage.rbd.csi.ceph.com**	Provisions a volume for RWO access mode for Ceph RBD, RWO and RWX access mode for block PVC, and RWO access mode for Filesystem PVC.
GCE Persistent Disk (gcePD)	**kubernetes.io/gce-pd**	In multi-zone configurations, it is advisable to run one OpenShift Container Platform cluster per GCE project to avoid PVs from being created in zones where no node in the current cluster exists.
S3 Bucket (MCG Object Bucket Claim)	**openshift-storage.noobaa.io/obc**	Provisions an object bucket claim to support S3 API calls through the Multicloud Object Gateway (MCG). The exact storage backing the S3 bucket is dependent on the MCG configuration and the type of deployment.

Storage type	Provisioner plug-in name	Notes
VMware vSphere	**kubernetes.io/vsphere-volume**	

IMPORTANT

Any chosen provisioner plug-in also requires configuration for the relevant cloud, host, or third-party provider as per the relevant documentation.

CHAPTER 6. MULTICLOUD OBJECT GATEWAY

6.1. ABOUT THE MULTICLOUD OBJECT GATEWAY

The Multicloud Object Gateway (MCG) is a lightweight object storage service for OpenShift, allowing users to start small and then scale as needed on-premise, in multiple clusters, and with cloud-native storage.

6.2. ACCESSING THE MULTICLOUD OBJECT GATEWAY WITH YOUR APPLICATIONS

You can access the object service with any application targeting AWS S3 or code that uses AWS S3 Software Development Kit (SDK). Applications need to specify the MCG endpoint, an access key, and a secret access key. You can use your terminal or the MCG CLI to retrieve this information.

Prerequisites

- A running OpenShift Container Storage Platform
- Download the MCG command-line interface for easier management:

  ```
  # subscription-manager repos --enable=rh-ocs-4-for-rhel-8-x86_64-rpms
  # yum install mcg
  ```

You can access the relevant endpoint, access key, and secret access key two ways:

- Section 6.2.1, “Accessing the Multicloud Object Gateway from the terminal”
- Section 6.2.2, “Accessing the Multicloud Object Gateway from the MCG command-line interface”

6.2.1. Accessing the Multicloud Object Gateway from the terminal

Procedure

Run the **describe** command to view information about the MCG endpoint, including its access key (**AWS_ACCESS_KEY_ID** value) and secret access key (**AWS_SECRET_ACCESS_KEY** value):

```
# oc describe noobaa -n openshift-storage
```

The output will look similar to the following:

```
Name:         noobaa
Namespace:    openshift-storage
Labels:       <none>
Annotations:  <none>
API Version:  noobaa.io/v1alpha1
Kind:         NooBaa
Metadata:
  Creation Timestamp:  2019-07-29T16:22:06Z
  Generation:          1
  Resource Version:    6718822
  Self Link:           /apis/noobaa.io/v1alpha1/namespaces/openshift-storage/noobaas/noobaa
```

```
  UID:               019cfb4a-b21d-11e9-9a02-06c8de012f9e
Spec:
Status:
  Accounts:
    Admin:
      Secret Ref:
        Name:           noobaa-admin
        Namespace:      openshift-storage
  Actual Image:         noobaa/noobaa-core:4.0
  Observed Generation:  1
  Phase:                Ready
  Readme:

  Welcome to NooBaa!
  -----------------

  Welcome to NooBaa!
    -----------------
    NooBaa Core Version:
    NooBaa Operator Version:

    Lets get started:

    1. Connect to Management console:

      Read your mgmt console login information (email & password) from secret: "noobaa-admin".

        kubectl get secret noobaa-admin -n openshift-storage -o json | jq '.data|map_values(@base64d)'

      Open the management console service - take External IP/DNS or Node Port or use port forwarding:

        kubectl port-forward -n openshift-storage service/noobaa-mgmt 11443:443 &
        open https://localhost:11443

    2. Test S3 client:

      kubectl port-forward -n openshift-storage service/s3 10443:443 &
1
      NOOBAA_ACCESS_KEY=$(kubectl get secret noobaa-admin -n openshift-storage -o json | jq -r '.data.AWS_ACCESS_KEY_ID|@base64d')
2
      NOOBAA_SECRET_KEY=$(kubectl get secret noobaa-admin -n openshift-storage -o json | jq -r '.data.AWS_SECRET_ACCESS_KEY|@base64d')
      alias s3='AWS_ACCESS_KEY_ID=$NOOBAA_ACCESS_KEY AWS_SECRET_ACCESS_KEY=$NOOBAA_SECRET_KEY aws --endpoint https://localhost:10443 --no-verify-ssl s3'
      s3 ls


    Services:
      Service Mgmt:
        External DNS:
          https://noobaa-mgmt-openshift-storage.apps.mycluster-cluster.qe.rh-ocs.com
          https://a3406079515be11eaa3b70683061451e-1194613580.us-east-2.elb.amazonaws.com:443
      Internal DNS:
        https://noobaa-mgmt.openshift-storage.svc:443
      Internal IP:
        https://172.30.235.12:443
      Node Ports:
        https://10.0.142.103:31385
      Pod Ports:
        https://10.131.0.19:8443
    serviceS3:
      External DNS: 3
        https://s3-openshift-storage.apps.mycluster-cluster.qe.rh-ocs.com
        https://a340f4e1315be11eaa3b70683061451e-943168195.us-east-2.elb.amazonaws.com:443
      Internal DNS:
        https://s3.openshift-storage.svc:443
      Internal IP:
        https://172.30.86.41:443
      Node Ports:
        https://10.0.142.103:31011
      Pod Ports:
        https://10.131.0.19:6443
```

1. access key (**AWS_ACCESS_KEY_ID** value)
2. secret access key (**AWS_SECRET_ACCESS_KEY** value)
3. MCG endpoint

NOTE

The output from the **oc describe noobaa** command lists the internal and external DNS names that are available. When using the internal DNS, the traffic is free. The external DNS uses Load Balancing to process the traffic, and therefore has a cost per hour.

6.2.2. Accessing the Multicloud Object Gateway from the MCG command-line interface

Prerequisites

- Download the MCG command-line interface:

  ```
  # subscription-manager repos --enable=rh-ocs-4-for-rhel-8-x86_64-rpms
  # yum install mcg
  ```

Procedure

Run the **status** command to access the endpoint, access key, and secret access key:

```
noobaa status -n openshift-storage
```

The output will look similar to the following:

```
INFO[0000] Namespace: openshift-storage
INFO[0000]
```

```
INFO[0000] CRD Status:
INFO[0003]   Exists: CustomResourceDefinition "noobaas.noobaa.io"
INFO[0003]   Exists: CustomResourceDefinition "backingstores.noobaa.io"
INFO[0003]   Exists: CustomResourceDefinition "bucketclasses.noobaa.io"
INFO[0004]   Exists: CustomResourceDefinition "objectbucketclaims.objectbucket.io"
INFO[0004]   Exists: CustomResourceDefinition "objectbuckets.objectbucket.io"
INFO[0004]
INFO[0004] Operator Status:
INFO[0004]   Exists: Namespace "openshift-storage"
INFO[0004]   Exists: ServiceAccount "noobaa"
INFO[0005]   Exists: Role "ocs-operator.v0.0.271-6g45f"
INFO[0005]   Exists: RoleBinding "ocs-operator.v0.0.271-6g45f-noobaa-f9vpj"
INFO[0006]   Exists: ClusterRole "ocs-operator.v0.0.271-fjhgh"
INFO[0006]   Exists: ClusterRoleBinding "ocs-operator.v0.0.271-fjhgh-noobaa-pdxn5"
INFO[0006]   Exists: Deployment "noobaa-operator"
INFO[0006]
INFO[0006] System Status:
INFO[0007]   Exists: NooBaa "noobaa"
INFO[0007]   Exists: StatefulSet "noobaa-core"
INFO[0007]   Exists: Service "noobaa-mgmt"
INFO[0008]   Exists: Service "s3"
INFO[0008]   Exists: Secret "noobaa-server"
INFO[0008]   Exists: Secret "noobaa-operator"
INFO[0008]   Exists: Secret "noobaa-admin"
INFO[0009]   Exists: StorageClass "openshift-storage.noobaa.io"
INFO[0009]   Exists: BucketClass "noobaa-default-bucket-class"
INFO[0009]   (Optional) Exists: BackingStore "noobaa-default-backing-store"
INFO[0010]   (Optional) Exists: CredentialsRequest "noobaa-cloud-creds"
INFO[0010]   (Optional) Exists: PrometheusRule "noobaa-prometheus-rules"
INFO[0010]   (Optional) Exists: ServiceMonitor "noobaa-service-monitor"
INFO[0011]   (Optional) Exists: Route "noobaa-mgmt"
INFO[0011]   (Optional) Exists: Route "s3"
INFO[0011]   Exists: PersistentVolumeClaim "db-noobaa-core-0"
INFO[0011]   System Phase is "Ready"
INFO[0011]   Exists:  "noobaa-admin"

#------------------#
#- Mgmt Addresses -#
#------------------#

1
ExternalDNS : [https://noobaa-mgmt-openshift-storage.apps.mycluster-cluster.qe.rh-ocs.com
https://a3406079515be11eaa3b70683061451e-1194613580.us-east-2.elb.amazonaws.com:443]
ExternalIP  : []
NodePorts   : [https://10.0.142.103:31385]
InternalDNS : [https://noobaa-mgmt.openshift-storage.svc:443]
InternalIP  : [https://172.30.235.12:443]
PodPorts    : [https://10.131.0.19:8443]

#--------------------#
#- Mgmt Credentials -#
#--------------------#

email    : admin@noobaa.io
password : HKLbH1rSuVU0I/souIkSiA==
#----------------#
#- S3 Addresses -#
#----------------#

ExternalDNS : [https://s3-openshift-storage.apps.mycluster-cluster.qe.rh-ocs.com
https://a340f4e1315be11eaa3b70683061451e-943168195.us-east-2.elb.amazonaws.com:443]
ExternalIP  : []
NodePorts   : [https://10.0.142.103:31011]
InternalDNS : [https://s3.openshift-storage.svc:443]
InternalIP  : [https://172.30.86.41:443]
PodPorts    : [https://10.131.0.19:6443]

#------------------#
#- S3 Credentials -#
#------------------#

2
AWS_ACCESS_KEY_ID     : jVmAsu9FsvRHYmfjTiHV
3
AWS_SECRET_ACCESS_KEY : E//420VNedJfATvVSmDz6FMtsSAzuBv6z180PT5c

#------------------#
#- Backing Stores -#
#------------------#

NAME                          TYPE     TARGET-BUCKET                                              PHASE   AGE
noobaa-default-backing-store   aws-s3   noobaa-backing-store-15dc896d-7fe0-4bed-9349-
5942211b93c9   Ready   141h35m32s

#------------------#
#- Bucket Classes -#
#------------------#

NAME                         PLACEMENT                                                  PHASE   AGE
noobaa-default-bucket-class   {Tiers:[{Placement: BackingStores:[noobaa-default-backing-store]}]}
Ready   141h35m33s

#-----------------#
#- Bucket Claims -#
#-----------------#

No OBC's found.
```

1 endpoint

2 access key

3 secret access key

You now have the relevant endpoint, access key, and secret access key in order to connect to your applications.

Example 6.1. Example

If AWS S3 CLI is the application, the following command will list buckets in OCS:

```
AWS_ACCESS_KEY_ID=<AWS_ACCESS_KEY_ID>
AWS_SECRET_ACCESS_KEY=<AWS_SECRET_ACCESS_KEY>
aws --endpoint <ENDPOINT> -- no-verify-ssl s3 ls
```

6.3. ADDING STORAGE RESOURCES FOR HYBRID OR MULTICLOUD

6.3.1. Adding storage resources for hybrid or Multicloud using the MCG command line interface

The Multicloud Object Gateway (MCG) simplifies the process of spanning data across cloud provider and clusters.

To do so, add a backing storage that can be used by the MCG.

For VMWare deployments, skip to Section 6.3.2, “Creating an S3 compatible NooBaa backingstore” for further instructions.

Prerequisites

- Download the MCG command-line interface:

  ```
  # subscription-manager repos --enable=rh-ocs-4-for-rhel-8-x86_64-rpms
  # yum install mcg
  ```

Procedure

1. From the MCG command-line interface, run the following command:

   ```
   noobaa backingstore create <backing-store-type> <backingstore_name> --access-key=
   <AWS ACCESS KEY> --secret-key=<AWS SECRET ACCESS KEY> --target-bucket
   <bucket-name>
   ```

   a. Replace **<backing-store-type>** with your relevant backing store type: **aws-s3**, **google-cloud-store**, **azure-blob**, or **s3-compatible**.

   b. Replace **<backingstore_name>** with the name of the backingstore.

   c. Replace **<AWS ACCESS KEY>** and **<AWS SECRET ACCESS KEY>** with an AWS access key ID and secret access key you created for this purpose.

   d. Replace **<bucket-name>** with an existing AWS bucket name. This argument tells NooBaa which bucket to use as a target bucket for its backing store, and subsequently, data storage and administration.
   The output will be similar to the following:

   ```
   INFO[0001]   Exists: NooBaa "noobaa"
   INFO[0002]   Created: BackingStore "aws-resource"
   INFO[0002]   Created: Secret "backing-store-secret-aws-resource"
   ```

You can also add storage resources using a YAML:

1. Create a secret with the credentials:

   ```
   apiVersion: v1
   kind: Secret
   metadata:
     name: <backingstore-secret-name>
   type: Opaque
   data:
     AWS_ACCESS_KEY_ID: <AWS ACCESS KEY ID ENCODED IN BASE64>
     AWS_SECRET_ACCESS_KEY: <AWS SECRET ACCESS KEY ENCODED IN BASE64>
   ```

   a. You must supply and encode your own AWS access key ID and secret access key using Base64, and use the results in place of **<AWS ACCESS KEY ID ENCODED IN BASE64>** and **<AWS SECRET ACCESS KEY ENCODED IN BASE64>**.

   b. Replace **<backingstore-secret-name>** with a unique name.

2. Apply the following YAML for a specific backing store:

   ```
   apiVersion: noobaa.io/v1alpha1
   kind: BackingStore
   metadata:
     finalizers:
     - noobaa.io/finalizer
     labels:
       app: noobaa
     name: bs
     namespace: noobaa
   spec:
     awsS3:
       secret:
         name: <backingstore-secret-name>
         namespace: noobaa
       targetBucket: <bucket-name>
     type: <backing-store-type>
   ```

   a. Replace **<bucket-name>** with an existing AWS bucket name. This argument tells NooBaa which bucket to use as a target bucket for its backing store, and subsequently, data storage and administration.

   b. Replace **<backingstore-secret-name>** with the name of the secret created in the previous step.

   c. Replace <backing-store-type> with your relevant backing store type: **aws-s3**, **google-cloud-store**, **azure-blob**, or **s3-compatible**.

6.3.2. Creating an S3 compatible NooBaa backingstore

Procedure

1. From the MCG command-line interface, run the following command:

   ```
   noobaa backingstore create s3-compatible rgw-resource --access-key=<RGW ACCESS
   KEY> --secret-key=<RGW SECRET KEY> --target-bucket=<bucket-name> --
   endpoint=http://rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-
   ```

```
storage.svc.cluster.local:80
```

a. To get the **<RGW ACCESS KEY>** and **<RGW SECRET KEY>**, run the following command using your RGW user secret name:

```
oc get secret <RGW USER SECRET NAME> -o yaml
```

b. Decode the access key ID and the access key from Base64 and keep them.

c. Replace **<RGW USER ACCESS KEY>** and **<RGW USER SECRET ACCESS KEY>** with the appropriate, decoded data from the previous step.

d. Replace **<bucket-name>** with an existing RGW bucket name. This argument tells NooBaa which bucket to use as a target bucket for its backing store, and subsequently, data storage and administration.
The output will be similar to the following:

```
INFO[0001]   Exists: NooBaa "noobaa"
INFO[0002]   Created: BackingStore "rgw-resource"
INFO[0002]   Created: Secret "backing-store-secret-rgw-resource"
```

You can also create the backingstore using a YAML:

1. Create a **CephObjectStore** user. This also creates a secret containing the RGW credentials:

```
apiVersion: ceph.rook.io/v1
kind: CephObjectStoreUser
metadata:
  name: <RGW-Username>
  namespace: openshift-storage
spec:
  store: ocs-storagecluster-cephobjectstore
  displayName: "<Display-name>"
```

a. Replace **<RGW-Username>** and **<Display-name>** with a unique username and display name.

2. Apply the following YAML for an S3-Compatible backing store:

```
apiVersion: noobaa.io/v1alpha1
kind: BackingStore
metadata:
  finalizers:
  - noobaa.io/finalizer
  labels:
    app: noobaa
  name: <backingstore-name>
  namespace: openshift-storage
spec:
  s3Compatible:
    endpoint: http://rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-
storage.svc.cluster.local:80
    secret:
      name: <backingstore-secret-name>
      namespace: openshift-storage
    signatureVersion: v4
    targetBucket: <RGW-bucket-name>
  type: s3-compatible
```

a. Replace **<backingstore-secret-name>** with the name of the secret that was created with **CephObjectStore** in the previous step.

b. Replace **<bucket-name>** with an existing RGW bucket name. This argument tells NooBaa which bucket to use as a target bucket for its backing store, and subsequently, data storage and administration.

6.4. MIRRORING DATA FOR HYBRID AND MULTICLOUD BUCKETS

The Multicloud Object Gateway (MCG) simplifies the process of spanning data across cloud provider and clusters.

Prerequisites

- You must first add a backing storage that can be used by the MCG, see Section 6.3, "Adding storage resources for hybrid or Multicloud".

Then you create a bucket class that reflects the data management policy, mirroring.

Procedure

You can set up mirroring data three ways:

- Section 6.4.1, "Creating bucket classes to mirror data using the MCG command-line-interface"
- Section 6.4.2, "Creating bucket classes to mirror data using a YAML"
- Section 6.4.3, "Configuring buckets to mirror data using the user interface"

6.4.1. Creating bucket classes to mirror data using the MCG command-line-interface

1. From the MCG command-line interface, run the following command to create a bucket class with a mirroring policy:

```
$ noobaa bucketclass create mirror-to-aws --backingstores=azure-resource,aws-resource --
placement Mirror
```

2. Set the newly created bucket class to a new bucket claim, generating a new bucket that will be mirrored between two locations:

```
$ noobaa obc create  mirrored-bucket --bucketclass=mirror-to-aws
```

6.4.2. Creating bucket classes to mirror data using a YAML

1. Apply the following YAML. This YAML is a hybrid example that mirrors data between local Ceph storage and AWS:

```
apiVersion: noobaa.io/v1alpha1
kind: BucketClass
```

```
metadata:
  name: hybrid-class
  labels:
    app: noobaa
spec:
  placementPolicy:
    tiers:
      - tier:
          mirrors:
            - mirror:
                spread:
                  - cos-east-us
            - mirror:
                spread:
                  - noobaa-test-bucket-for-ocp201907291921-11247_resource
```

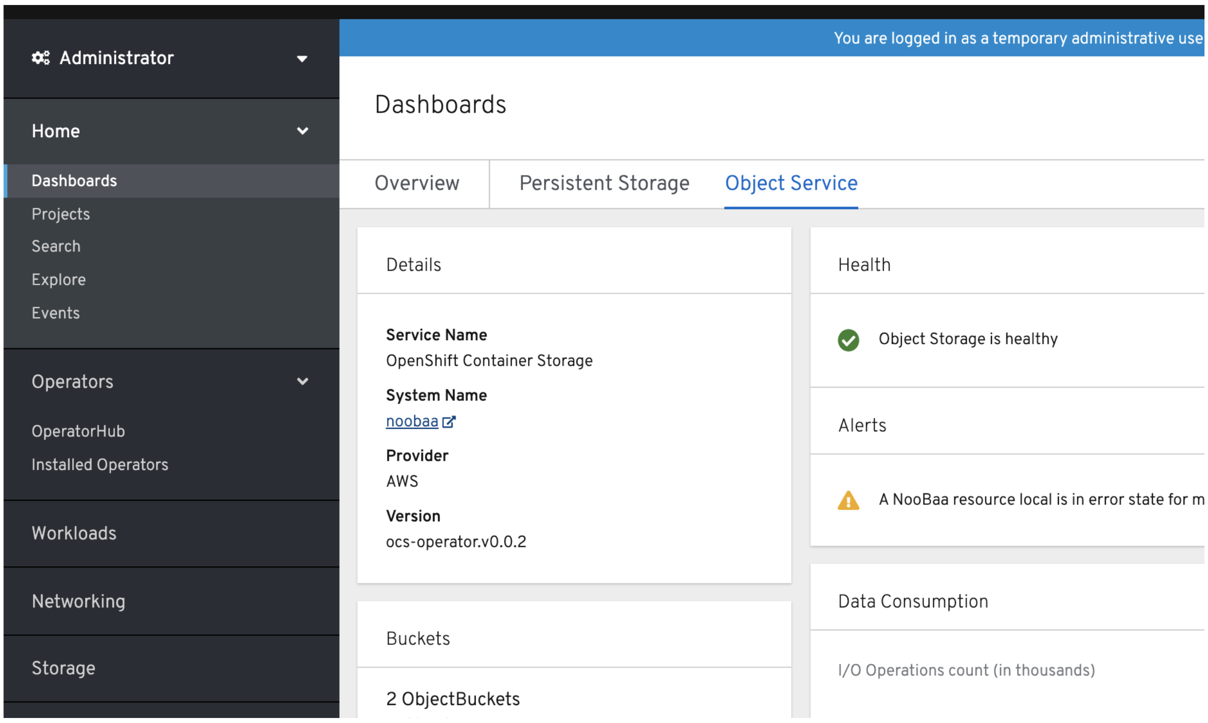
2. Add the following lines to your standard Object Bucket Claim (OBC):

```
additionalConfig:
  bucketclass: mirror-to-aws
```

For more information about OBCs, see Section 2.4, “Object Bucket Claim” .

6.4.3. Configuring buckets to mirror data using the user interface

1. In your OpenShift Storage console, navigate to **Dashboards** → **Object Service** → select the **noobaa** link:



You are logged in as a temporary administrative use

Dashboards

Service Name
OpenShift Container Storage

System Name
noobaa

Provider
AWS

Version
ocs-operator.v0.0.2

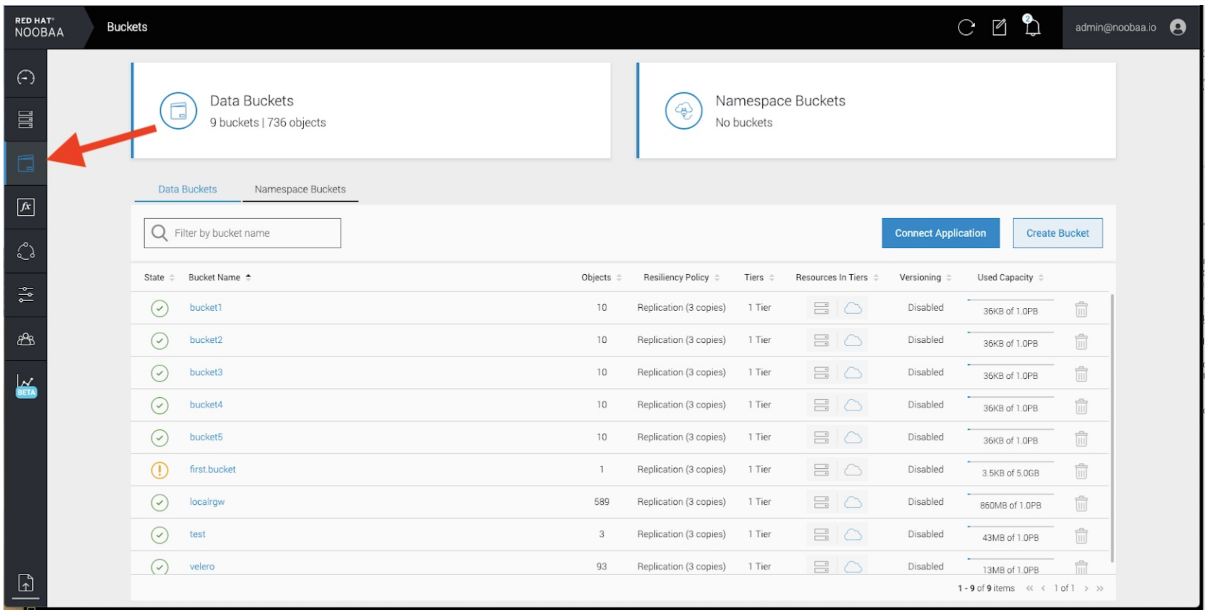
Object Storage is healthy

A NooBaa resource local is in error state for m

I/O Operations count (in thousands)

2 ObjectBuckets

2. Click the **buckets** icon on the left side. You will see a list of your buckets:



Data Buckets
9 buckets | 736 objects

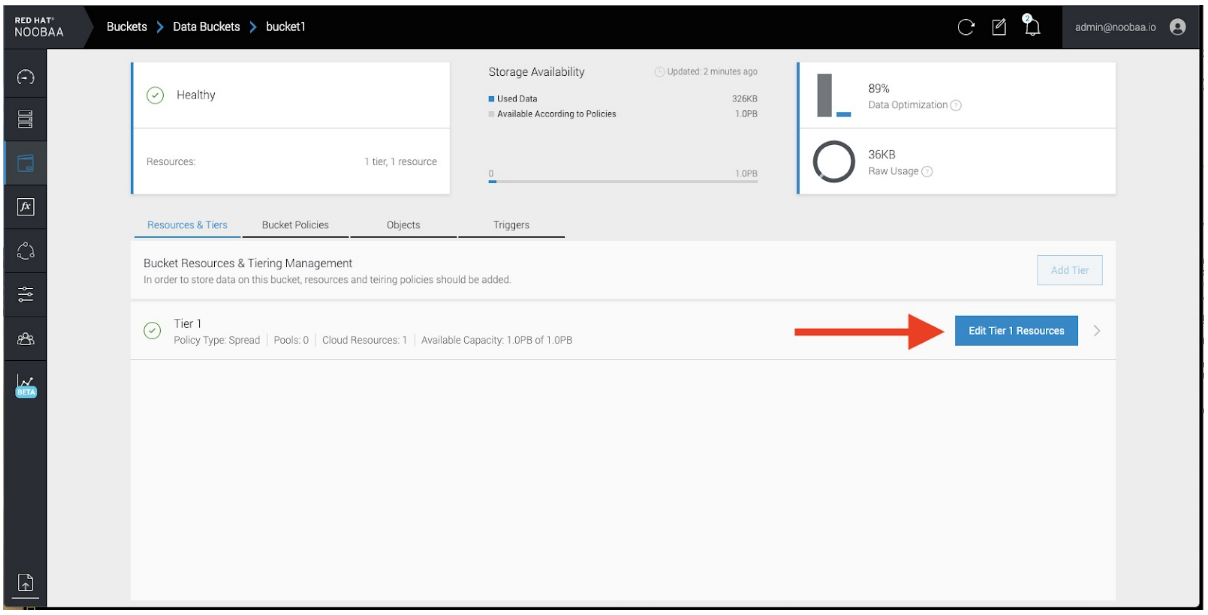
Namespace Buckets
No buckets

State	Bucket Name	Objects	Resiliency Policy	Tiers	Resources In Tiers	Versioning	Used Capacity
✓	bucket1	10	Replication (3 copies)	1 Tier		Disabled	36KB of 1.0PB
✓	bucket2	10	Replication (3 copies)	1 Tier		Disabled	36KB of 1.0PB
✓	bucket3	10	Replication (3 copies)	1 Tier		Disabled	36KB of 1.0PB
✓	bucket4	10	Replication (3 copies)	1 Tier		Disabled	36KB of 1.0PB
✓	bucket5	10	Replication (3 copies)	1 Tier		Disabled	36KB of 1.0PB
!	first.bucket	1	Replication (3 copies)	1 Tier		Disabled	3.5KB of 5.0GB
✓	localrgw	589	Replication (3 copies)	1 Tier		Disabled	860MB of 1.0PB
✓	test	3	Replication (3 copies)	1 Tier		Disabled	43MB of 1.0PB
✓	velero	93	Replication (3 copies)	1 Tier		Disabled	13MB of 1.0PB

1 - 9 of 9 items 1 of 1

3. Cick the bucket you want to update.

4. Click **Edit Tier 1 Resources**:



Healthy

Resources: 1 tier, 1 resource

Storage Availability Updated: 2 minutes ago

Used Data	326KB
Available According to Policies	1.0PB

0 1.0PB

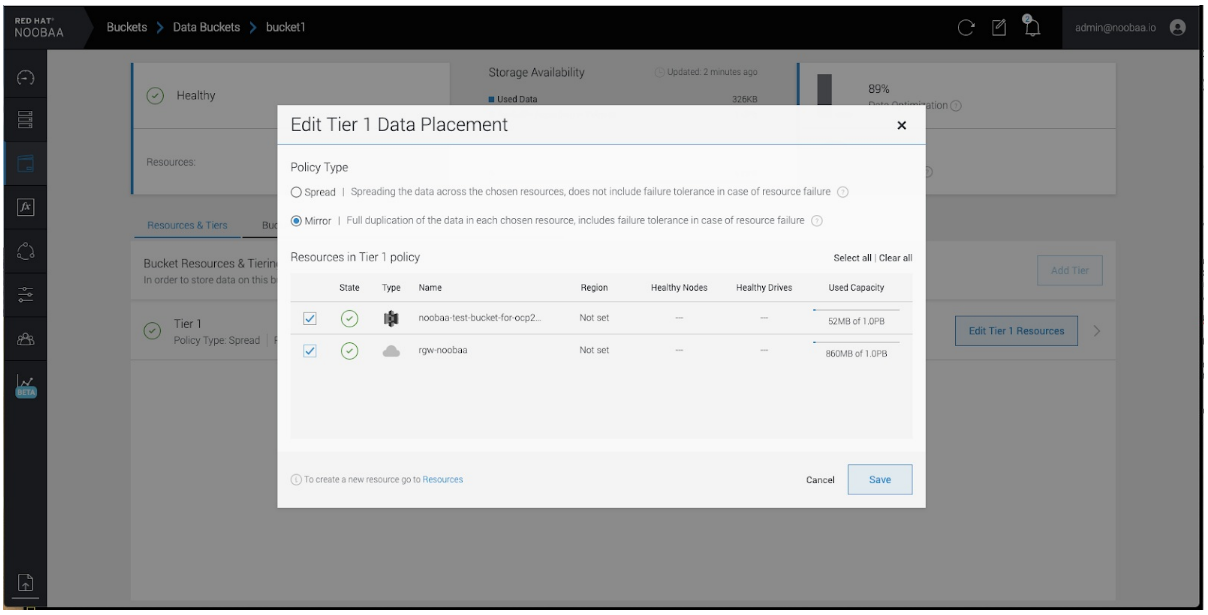
89%
Data Optimization

36KB
Raw Usage

Bucket Resources & Tiering Management
In order to store data on this bucket, resources and teiring policies should be added.

Tier 1
Policy Type: Spread | Pools: 0 | Cloud Resources: 1 | Available Capacity: 1.0PB of 1.0PB

5. Select **Mirror** and check the relevant resources you want to use for this bucket. In the following example, we mirror data between on prem Ceph RGW to AWS:



Edit Tier 1 Data Placement

Policy Type

( ) Spread | Spreading the data across the chosen resources, does not include failure tolerance in case of resource failure

[✓ Mirror] | Full duplication of the data in each chosen resource, includes failure tolerance in case of resource failure

Resources in Tier 1 policy — Select all | Clear all

	State	Type	Name	Region	Healthy Nodes	Healthy Drives	Used Capacity
✓	✓		noobaa-test-bucket-for-ocp2...	Not set	---	---	52MB of 1.0PB
✓	✓		rgw-noobaa	Not set	---	---	860MB of 1.0PB

To create a new resource go to Resources

Cancel | Save

6. Click **Save**.

NOTE

Resources created in NooBaa UI cannot be used by OpenShift UI or MCG CLI.

CHAPTER 7. REPLACING STORAGE NODES FOR OPENSHIFT CONTAINER STORAGE

For OpenShift Container Storage 4.2, node replacement can be performed proactively for an operational node and reactively for a failed node for the following deployments:

- For Amazon Web Services (AWS)
  - User-provisioned infrastructure
  - Installer-provisioned infrastructure
- For VMware
  - User-provisioned infrastructure

7.1. OPENSHIFT CONTAINER STORAGE DEPLOYED ON AWS

7.1.1. Replacing an operational AWS node on user-provisioned infrastructure

Perform this procedure to replace an operational node on AWS user-provisioned infrastructure.

Procedure

1. Identify the node that needs to be replaced.
2. Mark the node as unschedulable using the following command:

   ```
   $ oc adm cordon <node_name>
   ```

3. Drain the node using the following command:

   ```
   $ oc adm drain <node_name> --force --delete-local-data --ignore-daemonsets
   ```

   IMPORTANT

   This activity may take at least 5-10 minutes or more. Ceph errors generated during this period are temporary and are automatically resolved when the new node is labeled and functional.

4. Delete the node using the following command:

   ```
   $ oc delete nodes <node_name>
   ```

5. Create a new AWS machine instance with the required infrastructure. See Infrastructure requirements.
6. Create a new OpenShift Container Platform node using the new AWS machine instance.
7. Check for certificate signing requests (CSRs) related to OpenShift Container Platform that are in **Pending** state:

```
$ oc get csr
```

8. Approve all required OpenShift Container Platform CSRs for the new node:

```
$ oc adm certificate approve <Certificate_Name>
```

9. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.

10. Apply the OpenShift Container Storage label to the new node using any one of the following:

    From User interface

    a. For the new node, click **Action Menu (⋮) → Edit Labels**.

    b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

    From Command line interface

    - Execute the following command to apply the OpenShift Container Storage label to the new node:

```
$ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
```

11. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

```
$ oc delete pod rook-ceph-mgr-xxxx
```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

```
$ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:

    - **csi-cephfsplugin-***
    - **csi-rbdplugin-***

3. Verify that all other required OpenShift Container Storage pods are in **Running** state.

4. If verification steps fail, kindly contact Red Hat Support .

7.1.2. Replacing an operational AWS node on installer-provisioned infrastructure

Perform this procedure to replace an operational node on AWS installer-provisioned infrastructure (IPI).

Procedure

1. Log in to OpenShift Web Console and click **Compute → Nodes**.

2. Identify the node that needs to be replaced. Take a note of its **Machine Name**.

3. Mark the node as unschedulable using the following command:

```
$ oc adm cordon <node_name>
```

4. Drain the node using the following command:

```
$ oc adm drain <node_name> --force --delete-local-data --ignore-daemonsets
```

IMPORTANT

This activity may take at least 5-10 minutes or more. Ceph errors generated during this period are temporary and are automatically resolved when the new node is labeled and functional.

5. Click **Compute → Machines**. Search for the required machine.

6. Besides the required machine, click the **Action menu (⋮) → Delete Machine**.

7. Click **Delete** to confirm the machine deletion. A new machine is automatically created.

8. Wait for new machine to start and transition into **Running** state.

IMPORTANT

This activity may take at least 5-10 minutes or more.

9. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.

10. Apply the OpenShift Container Storage label to the new node using any one of the following:

    From User interface

    a. For the new node, click **Action Menu (⋮) → Edit Labels**

    b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

    From Command line interface

    - Execute the following command to apply the OpenShift Container Storage label to the new node:

    ```
    $ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
    ```

11. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

```
$ oc delete pod rook-ceph-mgr-xxxx
```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

```
$ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***
3. Verify that all other required OpenShift Container Storage pods are in **Running** state.
4. If verification steps fail, kindly contact Red Hat Support .

7.1.3. Replacing a failed AWS node on user-provisioned infrastructure

Perform this procedure to replace a failed node which is not operational on AWS user-provisioned infrastructure (UPI) for OpenShift Container Storage 4.2.

Procedure

1. Identify the AWS machine instance of the node that needs to be replaced.
2. Log in to AWS and terminate the identified AWS machine instance.
3. Create a new AWS machine instance with the required infrastructure. See Infrastructure requirements.
4. Create a new OpenShift Container Platform node using the new AWS machine instance.
5. Check for certificate signing requests (CSRs) related to OpenShift Container Platform that are in **Pending** state:

   ```
   $ oc get csr
   ```

6. Approve all required OpenShift Container Platform CSRs for the new node:

   ```
   $ oc adm certificate approve <Certificate_Name>
   ```

7. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.
8. Apply the OpenShift Container Storage label to the new node using any one of the following:

   From User interface

   a. For the new node, click **Action Menu (⋮) → Edit Labels**

   b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

   From Command line interface

   - Execute the following command to apply the OpenShift Container Storage label to the new node:

     ```
     $ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
     ```

9. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

```
$ oc delete pod rook-ceph-mgr-xxxx
```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

   ```
   $ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
   ```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***
3. Verify that all other required OpenShift Container Storage pods are in **Running** state.
4. If verification steps fail, kindly contact Red Hat Support .

7.1.4. Replacing a failed AWS node on installer-provisioned infrastructure

Perform this procedure to replace a failed node which is not operational on AWS installer-provisioned infrastructure (IPI) for OpenShift Container Storage 4.2.

Procedure

1. Log in to OpenShift Web Console and click **Compute → Nodes**.
2. Identify the faulty node and click on its **Machine Name**.
3. Click **Actions → Edit Annotations**, and click **Add More**.
4. Add **machine.openshift.io/exclude-node-draining** and click **Save**.
5. Click **Actions → Delete Machine**, and click **Delete**.
6. A new machine is automatically created, wait for new machine to start.

   IMPORTANT

   This activity may take at least 5-10 minutes or more. Ceph errors generated during this period are temporary and are automatically resolved when the new node is labeled and functional.

7. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.
8. Apply the OpenShift Container Storage label to the new node using any one of the following:

   From User interface

   a. For the new node, click **Action Menu (⋮) → Edit Labels**

   b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

   From Command line interface

- Execute the following command to apply the OpenShift Container Storage label to the new node:

  ```
  $ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
  ```

9. [Optional]: If the failed AWS instance is not removed automatically, terminate the instance from AWS console.

10. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

    ```
    $ oc delete pod rook-ceph-mgr-xxxx
    ```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

   ```
   $ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
   ```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***

3. Verify that all other required OpenShift Container Storage pods are in **Running** state.

4. If verification steps fail, kindly contact Red Hat Support .

7.2. OPENSHIFT CONTAINER STORAGE DEPLOYED ON VMWARE

7.2.1. Replacing an operational VMware node on user-provisioned infrastructure

Perform this procedure to replace an operational node on VMware user-provisioned infrastructure (UPI).

Procedure

1. Identify the node and its VM that needs to be replaced.

2. Mark the node as unschedulable using the following command:

   ```
   $ oc adm cordon <node_name>
   ```

3. Drain the node using the following command:

   ```
   $ oc adm drain <node_name> --force --delete-local-data --ignore-daemonsets
   ```

IMPORTANT

This activity may take at least 5-10 minutes or more. Ceph errors generated during this period are temporary and are automatically resolved when the new node is labeled and functional.

4. Delete the node using the following command:

```
$ oc delete nodes <node_name>
```

5. Log in to vSphere and terminate the identified VM.

IMPORTANT

VM should be deleted only from the inventory and not from the disk.

6. Create a new VM on vSphere with the required infrastructure. See Infrastructure requirements.

7. Create a new OpenShift Container Platform worker node using the new VM.

8. Check for certificate signing requests (CSRs) related to OpenShift Container Platform that are in **Pending** state:

```
$ oc get csr
```

9. Approve all required OpenShift Container Platform CSRs for the new node:

```
$ oc adm certificate approve <Certificate_Name>
```

10. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.

11. Apply the OpenShift Container Storage label to the new node using any one of the following:

From User interface

a. For the new node, click **Action Menu (⋮) → Edit Labels**

b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

From Command line interface

- Execute the following command to apply the OpenShift Container Storage label to the new node:

```
$ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
```

12. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

```
$ oc delete pod rook-ceph-mgr-xxxx
```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

```
$ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***
3. Verify that all other required OpenShift Container Storage pods are in **Running** state.
4. If verification steps fail, kindly contact Red Hat Support .

7.2.2. Replacing a failed VMware node on user-provisioned infrastructure

Perform this procedure to replace a failed node on VMware user-provisioned infrastructure (UPI).

Procedure

1. Identify the node and its VM that needs to be replaced.
2. Delete the node using the following command:

```
$ oc delete nodes <node_name>
```

3. Log in to vSphere and terminate the identified VM.

IMPORTANT

VM should be deleted only from the inventory and not from the disk.

4. Create a new VM on vSphere with the required infrastructure. See Infrastructure requirements.
5. Create a new OpenShift Container Platform worker node using the new VM.
6. Check for certificate signing requests (CSRs) related to OpenShift Container Platform that are in **Pending** state:

```
$ oc get csr
```

7. Approve all required OpenShift Container Platform CSRs for the new node:

```
$ oc adm certificate approve <Certificate_Name>
```

8. Click **Compute → Nodes**, confirm if the new node is in **Ready** state.
9. Apply the OpenShift Container Storage label to the new node using any one of the following:

   From User interface

   a. For the new node, click **Action Menu (⋮) → Edit Labels**

b. Add **cluster.ocs.openshift.io/openshift-storage** and click **Save**.

From Command line interface

- Execute the following command to apply the OpenShift Container Storage label to the new node:

```
$ oc label node <new_node_name> cluster.ocs.openshift.io/openshift-storage=""
```

10. Restart the **mgr** pod to update the OpenShift Container Storage with the new hostname.

```
$ oc delete pod rook-ceph-mgr-xxxx
```

Verification steps

1. Execute the following command and verify that the new node is present in the output:

```
$ oc get nodes --show-labels | grep cluster.ocs.openshift.io/openshift-storage= |cut -d' ' -f1
```

2. Click **Workloads → Pods**, confirm that at least the following pods on the new node are in **Running** state:
   - **csi-cephfsplugin-***
   - **csi-rbdplugin-***
3. Verify that all other required OpenShift Container Storage pods are in **Running** state.
4. If verification steps fail, kindly contact Red Hat Support .

CHAPTER 8. UPDATING OPENSHIFT CONTAINER STORAGE

8.1. ENABLING AUTOMATIC UPDATES FOR OPENSHIFT CONTAINER STORAGE OPERATOR

Use this procedure to enable automatic update approval for updating OpenShift Container Storage operator in OpenShift Container Platform.

Prerequisites

- Update the OpenShift Container Platform cluster to the latest stable release of version 4.2, see Updating Clusters.
- Ensure that all OpenShift Container Storage nodes are in **Ready** status.
- Under **Persistent Storage** in **Health** card, confirm that the Ceph cluster is healthy and data is resilient.

Procedure

1. Log in to OpenShift Web Console.
2. Click **Operators → Installed Operators**
3. Select the **openshift-storage** project.
4. Click on the OpenShift Container Storage operator name.
5. Click **Subscription** tab and click the link under **Approval**.
6. Select **Automatic (default)** and click **Save**.
7. Perform one of the following depending on the **Upgrade Status**:
   - Upgrade Status *shows* **requires approval**.
     a. Click on the **...requires approval**.
     b. On the **InstallPlan Details** page, click **Preview Install Plan**.
     c. Review the install plan and click **Approve**.
     d. Wait for the **Status** to change from **Unknown** to **Created**.
     e. Click **Operators → Installed Operators**
     f. Select the **openshift-storage** project.
     g. Wait for the **Status** to change to **Up to date**
   - Upgrade Status *does not show* **requires approval**:
     a. Wait for the update to initiate. This may take up to 20 minutes.
     b. Click **Operators → Installed Operators**

  c. Select the **openshift-storage** project.

  d. Wait for the **Status** to change to **Up to date**

Verification steps

1. Click **Persistent Storage** and in **Health** card confirm that the Ceph cluster is healthy.
2. Click **Operators → Installed Operators → OpenShift Container Storage Operator**.
3. Under **Storage Cluster**, verify that the cluster service status in **Ready**.
4. If verification steps fail, kindly contact Red Hat Support .

8.2. MANUALLY UPDATING OPENSHIFT CONTAINER STORAGE OPERATOR

Use this procedure to update OpenShift Container Storage operator by providing manual approval to the install plan.

Prerequisites

- Update the OpenShift Container Platform cluster to the latest stable release of version 4.2, see Updating Clusters.
- Ensure that all OpenShift Container Storage nodes are in **Ready** status.
- Under **Persistent Storage** in **Health** card, confirm that the Ceph cluster is healthy and data is resilient.

Procedure

1. Log in to OpenShift Web Console.
2. Click **Operators → Installed Operators**
3. Select the **openshift-storage** project.
4. Click **Subscription** tab and click the link under **Approval**.
5. Select **Manual** and click **Save**.
6. Wait for the **Upgrade Status** to change to **Upgrading**.
7. If the **Upgrade Status** shows **requires approval**, click on **requires approval**.
8. On the **InstallPlan Details** page, click **Preview Install Plan**.
9. Review the install plan and click **Approve**.
10. Wait for the **Status** to change from **Unknown** to **Created**.
11. Click **Operators → Installed Operators**
12. Select the **openshift-storage** project.

13. Wait for the **Status** to change to **Up to date**

Verification steps

1. Click **Persistent Storage** and in **Health** card confirm that the Ceph cluster is healthy.
2. Click **Operators → Installed Operators → OpenShift Container Storage Operator**.
3. Under **Storage Cluster**, verify that the cluster service status in **Ready**.
4. If verification steps fail, kindly contact Red Hat Support .