Red Hat OpenShift Data Foundation 4.15

Troubleshooting OpenShift Data Foundation

Instructions on troubleshooting OpenShift Data Foundation

Last Updated: 2024-03-19

Red Hat OpenShift Data Foundation 4.15 Troubleshooting OpenShift Data Foundation

Instructions on troubleshooting OpenShift Data Foundation

Legal Notice

Abstract

Read this document for instructions on troubleshooting Red Hat OpenShift Data Foundation.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

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

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

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

To give feedback, create a Bugzilla ticket:

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

CHAPTER 1. OVERVIEW

Troubleshooting OpenShift Data Foundation is written to help administrators understand how to troubleshoot and fix their Red Hat OpenShift Data Foundation cluster.

Most troubleshooting tasks focus on either a fix or a workaround. This document is divided into chapters based on the errors that an administrator may encounter:

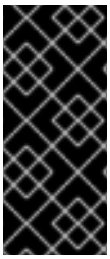
- Chapter 2, *Downloading log files and diagnostic information using must-gather* shows you how to use the must-gather utility in OpenShift Data Foundation.
- Chapter 3, *Commonly required logs for troubleshooting* shows you how to obtain commonly required log files for OpenShift Data Foundation.
- Chapter 6, *Troubleshooting alerts and errors in OpenShift Data Foundation* shows you how to identify the encountered error and perform required actions.

WARNING

Red Hat does not support running Ceph commands in OpenShift Data Foundation clusters (unless indicated by Red Hat support or Red Hat documentation) as it can cause data loss if you run the wrong commands. In that case, the Red Hat support team is only able to provide commercially reasonable effort and may not be able to restore all the data in case of any data loss.

CHAPTER 2. DOWNLOADING LOG FILES AND DIAGNOSTIC INFORMATION USING MUST-GATHER

If Red Hat OpenShift Data Foundation is unable to automatically resolve a problem, use the **must-gather** tool to collect log files and diagnostic information so that you or Red Hat support can review the problem and determine a solution.



IMPORTANT

When Red Hat OpenShift Data Foundation is deployed in external mode, **must-gather** only collects logs from the OpenShift Data Foundation cluster and does not collect debug data and logs from the external Red Hat Ceph Storage cluster. To collect debug logs from the external Red Hat Ceph Storage cluster, see Red Hat Ceph Storage Troubleshooting guide and contact your Red Hat Ceph Storage Administrator.

Prerequisites

- Optional: If OpenShift Data Foundation is deployed in a disconnected environment, ensure that you mirror the individual **must-gather** image to the mirror registry available from the disconnected environment.

```
$ oc image mirror registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 <local-registry>/odf4/odf-must-gather-rhel9:v4.15 [--registry-config=<path-to-the-registry-config>] [--insecure=true]
```

<local-registry>

Is the local image mirror registry available for a disconnected OpenShift Container Platform cluster.

<path-to-the-registry-config>

Is the path to your registry credentials, by default it is ~/**.docker/config.json**.

--insecure

Add this flag only if the mirror registry is insecure.

For more information, see the Red Hat Knowledgebase solutions:

- How to mirror images between Redhat Openshift registries
- Failed to mirror OpenShift image repository when private registry is insecure

Procedure

- Run the **must-gather** command from the client connected to the OpenShift Data Foundation cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 --dest-dir=<directory-name>
```

<directory-name>

Is the name of the directory where you want to write the data to.

IMPORTANT

For a disconnected environment deployment, replace the image in **--image** parameter with the mirrored **must-gather** image.

```
$ oc adm must-gather --image=<local-registry>/odf4/odf-must-gather-rhel9:v4.15 --dest-dir=<directory-name>
```

<local-registry>

Is the local image mirror registry available for a disconnected OpenShift Container Platform cluster.

This collects the following information in the specified directory:

- All Red Hat OpenShift Data Foundation cluster related Custom Resources (CRs) with their namespaces.
- Pod logs of all the Red Hat OpenShift Data Foundation related pods.
- Output of some standard Ceph commands like Status, Cluster health, and others.

2.1. VARIATIONS OF MUST-GATHER-COMMANDS

- If one or more master nodes are not in the **Ready** state, use **--node-name** to provide a master node that is **Ready** so that the **must-gather** pod can be safely scheduled.

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 --dest-dir=_<directory-name>_ --node-name=_<node-name>_
```

- If you want to gather information from a specific time:
  - To specify a relative time period for logs gathered, such as within 5 seconds or 2 days, add **/usr/bin/gather since=<duration>**:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 --dest-dir=_<directory-name>_ /usr/bin/gather since=<duration>
```

  - To specify a specific time to gather logs after, add **/usr/bin/gather since-time=<rfc3339-timestamp>**:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 --dest-dir=_<directory-name>_ /usr/bin/gather since-time=<rfc3339-timestamp>
```

Replace the example values in these commands as follows:

<node-name>

If one or more master nodes are not in the **Ready** state, use this parameter to provide the name of a master node that is still in the **Ready** state. This avoids scheduling errors by ensuring that the **must-gather** pod is not scheduled on a master node that is not ready.

<directory-name>

The directory to store information collected by **must-gather**.

<duration>

Specify the period of time to collect information from as a relative duration, for example, **5h** (starting from 5 hours ago).

<rfc3339-timestamp>

Specify the period of time to collect information from as an RFC 3339 timestamp, for example, **2020-11-10T04:00:00+00:00** (starting from 4 am UTC on 11 Nov 2020).

2.2. RUNNING MUST-GATHER IN MODULAR MODE

Red Hat OpenShift Data Foundation **must-gather** can take a long time to run in some environments. To avoid this, run **must-gather** in modular mode and collect only the resources you require using the following command:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15 -- /usr/bin/gather <-arg>
```

Replace **< -arg>** with one or more of the following arguments to specify the resources for which the must-gather logs is required.

-o, --odf

ODF logs (includes Ceph resources, namespaced resources, clusterscoped resources and Ceph logs)

-d, --dr

DR logs

-n, --noobaa

Noobaa logs

-c, --ceph

Ceph commands and pod logs

-cl, --ceph-logs

Ceph daemon, kernel, journal logs, and crash reports

-ns, --namespaced

namespaced resources

-cs, --clusterscoped

clusterscoped resources

-h, --help

Print help message

CHAPTER 3. COMMONLY REQUIRED LOGS FOR TROUBLESHOOTING

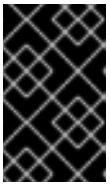
Some of the commonly used logs for troubleshooting OpenShift Data Foundation are listed, along with the commands to generate them.

- Generating logs for a specific pod:

```
$ oc logs <pod-name> -n <namespace>
```

- Generating logs for Ceph or OpenShift Data Foundation cluster:

```
$ oc logs rook-ceph-operator-<ID> -n openshift-storage
```



IMPORTANT

Currently, the rook-ceph-operator logs do not provide any information about the failure and this acts as a limitation in troubleshooting issues, see Enabling and disabling debug logs for rook-ceph-operator.

- Generating logs for plugin pods like cephfs or rbd to detect any problem in the PVC mount of the app-pod:

```
$ oc logs csi-cephfsplugin-<ID> -n openshift-storage -c csi-cephfsplugin
```

```
$ oc logs csi-rbdplugin-<ID> -n openshift-storage -c csi-rbdplugin
```

  - To generate logs for all the containers in the CSI pod:

```
$ oc logs csi-cephfsplugin-<ID> -n openshift-storage --all-containers
```

```
$ oc logs csi-rbdplugin-<ID> -n openshift-storage --all-containers
```

- Generating logs for cephfs or rbd provisioner pods to detect problems if PVC is not in **BOUND** state:

```
$ oc logs csi-cephfsplugin-provisioner-<ID> -n openshift-storage -c csi-cephfsplugin
```

```
$ oc logs csi-rbdplugin-provisioner-<ID> -n openshift-storage -c csi-rbdplugin
```

  - To generate logs for all the containers in the CSI pod:

```
$ oc logs csi-cephfsplugin-provisioner-<ID> -n openshift-storage --all-containers
```

```
$ oc logs csi-rbdplugin-provisioner-<ID> -n openshift-storage --all-containers
```

- Generating OpenShift Data Foundation logs using cluster-info command:

```
$ oc cluster-info dump -n openshift-storage --output-directory=<directory-name>
```

- When using Local Storage Operator, generating logs can be done using cluster-info command:

```
$ oc cluster-info dump -n openshift-local-storage --output-directory=<directory-name>
```

- Check the OpenShift Data Foundation operator logs and events.
  - To check the operator logs :

```
# oc logs <ocs-operator> -n openshift-storage
```

<ocs-operator>

```
# oc get pods -n openshift-storage | grep -i "ocs-operator" | awk '{print $1}'
```

  - To check the operator events :

```
# oc get events --sort-by=metadata.creationTimestamp -n openshift-storage
```

- Get the OpenShift Data Foundation operator version and channel.

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

Example output :

```
NAME                         DISPLAY                       VERSION   REPLACES   PHASE
mcg-operator.v4.15.0         NooBaa Operator               4.15.0               Succeeded
ocs-operator.v4.15.0         OpenShift Container Storage   4.15.0               Succeeded
odf-csi-addons-operator.v4.15.0   CSI Addons                    4.15.0               Succeeded
odf-operator.v4.15.0         OpenShift Data Foundation     4.15.0               Succeeded
```

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

Example output :

```
NAME                                              PACKAGE                SOURCE
CHANNEL
mcg-operator-stable-4.15-redhat-operators-openshift-marketplace   mcg-operator
redhat-operators   stable-4.15
ocs-operator-stable-4.15-redhat-operators-openshift-marketplace   ocs-operator
redhat-operators   stable-4.15
odf-csi-addons-operator                                odf-csi-addons-operator   redhat-operators
stable-4.15
odf-operator                                           odf-operator           redhat-operators   stable-
4.15
```

- Confirm that the **installplan** is created.

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

- Verify the image of the components post updating OpenShift Data Foundation.
  - Check the node on which the pod of the component you want to verify the image is running.

```
# oc get pods -o wide | grep <component-name>
```

For Example :

```
# oc get pods -o wide | grep rook-ceph-operator
```

Example output:

```
rook-ceph-operator-566cc677fd-bjqnb 1/1 Running 20 4h6m 10.128.2.5 rook-ceph-operator-566cc677fd-bjqnb 1/1 Running 20 4h6m 10.128.2.5 dell-r440-12.gsslab.pnq2.redhat.com <none> <none>

<none> <none>
```

dell-r440-12.gsslab.pnq2.redhat.com is the **node-name**.

- Check the image ID.

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

<node-name>

Is the name of the node on which the pod of the component you want to verify the image is running.

```
# chroot /host
```

```
# crictl images | grep <component>
```

For Example :

```
# crictl images | grep rook-ceph
```

Take a note of the **IMAGEID** and map it to the **Digest** ID on the Rook Ceph Operator page.

Additional resources

- Using must-gather

3.1. ADJUSTING VERBOSITY LEVEL OF LOGS

The amount of space consumed by debugging logs can become a significant issue. Red Hat OpenShift Data Foundation offers a method to adjust, and therefore control, the amount of storage to be consumed by debugging logs.

In order to adjust the verbosity levels of debugging logs, you can tune the log levels of the containers responsible for container storage interface (CSI) operations. In the container's yaml file, adjust the following parameters to set the logging levels:

- **CSI_LOG_LEVEL** - defaults to **5**
- **CSI_SIDECAR_LOG_LEVEL** - defaults to **1**

The supported values are **0** through **5**. Use **0** for general useful logs, and **5** for trace level verbosity.

CHAPTER 4. OVERRIDING THE CLUSTER-WIDE DEFAULT NODE SELECTOR FOR OPENSHIFT DATA FOUNDATION POST DEPLOYMENT

When a cluster-wide default node selector is used for OpenShift Data Foundation, the pods generated by container storage interface (CSI) daemonsets are able to start only on the nodes that match the selector. To be able to use OpenShift Data Foundation from nodes which do not match the selector, override the **cluster-wide default node selector** by performing the following steps in the command line interface :

Procedure

1. Specify a blank node selector for the openshift-storage namespace.

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

2. Delete the original pods generated by the DaemonSets.

```
oc delete pod -l app=csi-cephfsplugin -n openshift-storage
oc delete pod -l app=csi-rbdplugin -n openshift-storage
```

CHAPTER 5. ENCRYPTION TOKEN IS DELETED OR EXPIRED

Use this procedure to update the token if the encryption token for your key management system gets deleted or expires.

Prerequisites

- Ensure that you have a new token with the same policy as the deleted or expired token

Procedure

1. Log in to OpenShift Container Platform Web Console.
2. Click **Workloads → Secrets**
3. To update the **ocs-kms-token** used for cluster wide encryption:
    a. Set the **Project** to **openshift-storage**.
    b. Click **ocs-kms-token → Actions → Edit Secret**.
    c. Drag and drop or upload your encryption token file in the **Value** field. The token can either be a file or text that can be copied and pasted.
    d. Click **Save**.
4. To update the **ceph-csi-kms-token** for a given project or namespace with encrypted persistent volumes:
    a. Select the required **Project**.
    b. Click **ceph-csi-kms-token → Actions → Edit Secret**.
    c. Drag and drop or upload your encryption token file in the **Value** field. The token can either be a file or text that can be copied and pasted.
    d. Click **Save**.

NOTE

The token can be deleted only after all the encrypted PVCs using the **ceph-csi-kms-token** have been deleted.

CHAPTER 6. TROUBLESHOOTING ALERTS AND ERRORS IN OPENSHIFT DATA FOUNDATION

6.1. RESOLVING ALERTS AND ERRORS

Red Hat OpenShift Data Foundation can detect and automatically resolve a number of common failure scenarios. However, some problems require administrator intervention.

To know the errors currently firing, check one of the following locations:

- **Observe → Alerting → Firing** option
- **Home → Overview → Cluster** tab
- **Storage → Data Foundation → Storage System →** *storage system* link in the pop up **→ Overview → Block and File** tab
- **Storage → Data Foundation → Storage System →** *storage system* link in the pop up **→ Overview → Object** tab

Copy the error displayed and search it in the following section to know its severity and resolution:

Name: **CephMonVersionMismatch**

Message: **There are multiple versions of storage services running.**

Description: **There are {{ $value }} different versions of Ceph Mon components running.**

Severity: Warning

Resolution: Fix

Procedure: Inspect the user interface and log, and verify if an update is in progress.

- If an update is in progress, this alert is temporary.
- If an update is not in progress, restart the upgrade process.

Name: **CephOSDVersionMismatch**

Message: **There are multiple versions of storage services running.**

Description: **There are {{ $value }} different versions of Ceph OSD components running.**

Severity: Warning

Resolution: Fix

Procedure: Inspect the user interface and log, and verify if an update is in progress.

- If an update is in progress, this alert is temporary.
- If an update is not in progress, restart the upgrade process.

Name: **CephClusterCriticallyFull**

Message: **Storage cluster is critically full and needs immediate expansion**

Description: **Storage cluster utilization has crossed 85%.**

Severity: Crtical

Resolution: Fix

Procedure: Remove unnecessary data or expand the cluster.

Name: **CephClusterNearFull**

Fixed: **Storage cluster is nearing full. Expansion is required.**

Description: **Storage cluster utilization has crossed 75%.**

Severity: Warning

Resolution: Fix

Procedure: Remove unnecessary data or expand the cluster.

Name: **NooBaaBucketErrorState**

Message: **A NooBaa Bucket Is In Error State**

Description: **A NooBaa bucket {{ $labels.bucket_name }} is in error state for more than 6m**

Severity: Warning

Resolution: Workaround

Procedure: Resolving NooBaa Bucket Error State

Name: **NooBaaNamespaceResourceErrorState**

Message: **A NooBaa Namespace Resource Is In Error State**

Description: **A NooBaa namespace resource {{ $labels.namespace_resource_name }} is in error state for more than 5m**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Error State

Name: **NooBaaNamespaceBucketErrorState**

Message: **A NooBaa Namespace Bucket Is In Error State**

Description: **A NooBaa namespace bucket {{ $labels.bucket_name }} is in error state for more than 5m**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Error State

Name: **NooBaaBucketExceedingQuotaState**

Message: **A NooBaa Bucket Is In Exceeding Quota State**

Description: **A NooBaa bucket {{ $labels.bucket_name }} is exceeding its quota - {{ printf "%0.0f" $value }}% used message: A NooBaa Bucket Is In Exceeding Quota State**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Exceeding Quota State

Name: **NooBaaBucketLowCapacityState**

Message: **A NooBaa Bucket Is In Low Capacity State**

Description: **A NooBaa bucket {{ $labels.bucket_name }} is using {{ printf "%0.0f" $value }}% of its capacity**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **NooBaaBucketNoCapacityState**

Message: **A NooBaa Bucket Is In No Capacity State**

Description: **A NooBaa bucket {{ $labels.bucket_name }} is using all of its capacity**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **NooBaaBucketReachingQuotaState**

Message: **A NooBaa Bucket Is In Reaching Quota State**

Description: **A NooBaa bucket {{ $labels.bucket_name }} is using {{ printf "%0.0f" $value }}% of its quota**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **NooBaaResourceErrorState**

Message: **A NooBaa Resource Is In Error State**

Description: **A NooBaa resource {{ $labels.resource_name }} is in error state for more than 6m**

Severity: Warning

Resolution: Workaround

Procedure: Resolving NooBaa Bucket Error State

Name: **NooBaaSystemCapacityWarning100**

Message: **A NooBaa System Approached Its Capacity**

Description: **A NooBaa system approached its capacity, usage is at 100%**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **NooBaaSystemCapacityWarning85**

Message: **A NooBaa System Is Approaching Its Capacity**

Description: **A NooBaa system is approaching its capacity, usage is more than 85%**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **NooBaaSystemCapacityWarning95**

Message: **A NooBaa System Is Approaching Its Capacity**

Description: **A NooBaa system is approaching its capacity, usage is more than 95%**

Severity: Warning

Resolution: Fix

Procedure: Resolving NooBaa Bucket Capacity or Quota State

Name: **CephMdsMissingReplicas**

Message: **Insufficient replicas for storage metadata service.**

Description: `Minimum required replicas for storage metadata service not available.

Might affect the working of storage cluster.`

Severity: Warning

Resolution: Contact Red Hat support

Procedure:

1. Check for alerts and operator status.
2. If the issue cannot be identified, contact Red Hat support.

Name: **CephMgrIsAbsent**

Message: **Storage metrics collector service not available anymore.**

Description: **Ceph Manager has disappeared from Prometheus target discovery.**

Severity: Critical

Resolution: Contact Red Hat support

Procedure:

1. Inspect the user interface and log, and verify if an update is in progress.
   - If an update is in progress, this alert is temporary.
   - If an update is not in progress, restart the upgrade process.
2. Once the upgrade is complete, check for alerts and operator status.
3. If the issue persists or cannot be identified, contact Red Hat support.

Name: **CephNodeDown**

Message: **Storage node {{ $labels.node }} went down**

Description: **Storage node {{ $labels.node }} went down. Check the node immediately.**

Severity: Critical

Resolution: Contact Red Hat support

Procedure:

1. Check which node stopped functioning and its cause.
2. Take appropriate actions to recover the node. If node cannot be recovered:
   - See Replacing storage nodes for Red Hat OpenShift Data Foundation
   - Contact Red Hat support.

Name: **CephClusterErrorState**

Message: **Storage cluster is in error state**

Description: **Storage cluster is in error state for more than 10m.**

Severity: Critical

Resolution: Contact Red Hat support

Procedure:

1. Check for alerts and operator status.
2. If the issue cannot be identified, download log files and diagnostic information using must-gather.
3. Open a Support Ticket with Red Hat Support with an attachment of the output of must-gather.

Name: **CephClusterWarningState**

Message: **Storage cluster is in degraded state**

Description: **Storage cluster is in warning state for more than 10m.**

Severity: Warning

Resolution: Contact Red Hat support

Procedure:

1. Check for alerts and operator status.
2. If the issue cannot be identified, download log files and diagnostic information using must-gather.
3. Open a Support Ticket with Red Hat Support with an attachment of the output of must-gather.

Name: **CephDataRecoveryTakingTooLong**

Message: **Data recovery is slow**

Description: **Data recovery has been active for too long.**

Severity: Warning

Resolution: Contact Red Hat support

Name: **CephOSDDiskNotResponding**

Message: **Disk not responding**

Description: **Disk device {{ $labels.device }} not responding, on host {{ $labels.host }}.**

Severity: Critical

Resolution: Contact Red Hat support

Name: **CephOSDDiskUnavailable**

Message: **Disk not accessible**

Description: **Disk device {{ $labels.device }} not accessible on host {{ $labels.host }}.**

Severity: Critical

Resolution: Contact Red Hat support

Name: **CephPGRepairTakingTooLong**

Message: **Self heal problems detected**

Description: **Self heal operations taking too long.**

Severity: Warning

Resolution: Contact Red Hat support

Name: **CephMonHighNumberOfLeaderChanges**

Message: **Storage Cluster has seen many leader changes recently.**

Description: **'Ceph Monitor "{{ $labels.job }}": instance {{ $labels.instance }} has seen {{ $value printf "%.2f" }} leader changes per minute recently.'**

Severity: Warning

Resolution: Contact Red Hat support

Name: **CephMonQuorumAtRisk**

Message: **Storage quorum at risk**

Description: **Storage cluster quorum is low.**

Severity: Critical

Resolution: Contact Red Hat support

Name: **ClusterObjectStoreState**

Message: **Cluster Object Store is in an unhealthy state. Check Ceph cluster health**.

Description: **Cluster Object Store is in an unhealthy state for more than 15s. Check Ceph cluster health**.

Severity: Critical

Resolution: Contact Red Hat support

Procedure:

- Check the **CephObjectStore** CR instance.
- Contact Red Hat support.

Name: **CephOSDFlapping**

Message: **Storage daemon osd.x has restarted 5 times in the last 5 minutes. Check the pod events or Ceph status to find out the cause**.

Description: **Storage OSD restarts more than 5 times in 5 minutes**.

Severity: Critical

Resolution: Contact Red Hat support

Name: **OdfPoolMirroringImageHealth**

Message: **Mirroring image(s) (PV) in the pool <pool-name> are in Warning state for more than a 1m. Mirroring might not work as expected.**

Description: Disaster recovery is failing for one or a few applications.

Severity: Warning

Resolution: Contact Red Hat support

Name: **OdfMirrorDaemonStatus**

Message: **Mirror daemon is unhealthy**.

Description: Disaster recovery is failing for the entire cluster. Mirror daemon is in an unhealthy status for more than 1m. Mirroring on this cluster is not working as expected.

Severity: Critical

Resolution: Contact Red Hat support

6.2. RESOLVING CLUSTER HEALTH ISSUES

There is a finite set of possible health messages that a Red Hat Ceph Storage cluster can raise that show in the OpenShift Data Foundation user interface. These are defined as health checks which have unique identifiers. The identifier is a terse pseudo-human-readable string that is intended to enable tools to make sense of health checks, and present them in a way that reflects their meaning. Click the health code below for more information and troubleshooting.

Health code	Description
MON_DISK_LOW	One or more Ceph Monitors are low on disk space.

6.2.1. MON_DISK_LOW

This alert triggers if the available space on the file system storing the monitor database as a percentage, drops below **mon_data_avail_warn** (default: 15%). This may indicate that some other process or user on the system is filling up the same file system used by the monitor. It may also indicate that the monitor's database is large.

NOTE

The paths to the file system differ depending on the deployment of your mons. You can find the path to where the mon is deployed in **storagecluster.yaml**.

Example paths:

- Mon deployed over PVC path: **/var/lib/ceph/mon**
- Mon deployed over hostpath: **/var/lib/rook/mon**

In order to clear up space, view the high usage files in the file system and choose which to delete. To view the files, run:

```
# du -a <path-in-the-mon-node> |sort -n -r |head -n10
```

Replace **<path-in-the-mon-node>** with the path to the file system where mons are deployed.

6.3. RESOLVING CLUSTER ALERTS

There is a finite set of possible health alerts that a Red Hat Ceph Storage cluster can raise that show in

the OpenShift Data Foundation user interface. These are defined as health alerts which have unique identifiers. The identifier is a terse pseudo-human-readable string that is intended to enable tools to make sense of health checks, and present them in a way that reflects their meaning. Click the health alert for more information and troubleshooting.

Table 6.1. Types of cluster health alerts

Health alert	Overview
CephClusterCriticallyFull	Storage cluster utilization has crossed 80%.
CephClusterErrorState	Storage cluster is in an error state for more than 10 minutes.
CephClusterNearFull	Storage cluster is nearing full capacity. Data deletion or cluster expansion is required.
CephClusterReadOnly	Storage cluster is read-only now and needs immediate data deletion or cluster expansion.
CephClusterWarningState	Storage cluster is in a warning state for more than 10 mins.
CephDataRecoveryTakingTooLong	Data recovery has been active for too long.
CephMdsCacheUsageHigh	Ceph metadata service (MDS) cache usage for the MDS daemon has exceeded 95% of the **mds_cache_memory_limit**.
CephMdsCpuUsageHigh	Ceph MDS CPU usage for the MDS daemon has exceeded the threshold for adequate performance.
CephMdsMissingReplicas	Minimum required replicas for storage metadata service not available. Might affect the working of the storage cluster.
CephMgrIsAbsent	Ceph Manager has disappeared from Prometheus target discovery.
CephMgrIsMissingReplicas	Ceph manager is missing replicas. Thispts health status reporting and will cause some of the information reported by the **ceph status** command to be missing or stale. In addition, the Ceph manager is responsible for a manager framework aimed at expanding the existing capabilities of Ceph.
CephMonHighNumberOfLeaderChanges	The Ceph monitor leader is being changed an unusual number of times.
CephMonQuorumAtRisk	Storage cluster quorum is low.
CephMonQuorumLost	The number of monitor pods in the storage cluster are not enough.
CephMonVersionMismatch	There are different versions of Ceph Mon components running.

Health alert	Overview
CephNodeDown	A storage node went down. Check the node immediately. The alert should contain the node name.
CephOSDCriticallyFull	Utilization of back-end Object Storage Device (OSD) has crossed 80%. Free up some space immediately or expand the storage cluster or contact support.
CephOSDDiskNotResponding	A disk device is not responding on one of the hosts.
CephOSDDiskUnavailable	A disk device is not accessible on one of the hosts.
CephOSDFlapping	Ceph storage OSD flapping.
CephOSDNearFull	One of the OSD storage devices is nearing full.
CephOSDSlowOps	OSD requests are taking too long to process.
CephOSDVersionMismatch	There are different versions of Ceph OSD components running.
CephPGRepairTakingTooLong	Self-healing operations are taking too long.
CephPoolQuotaBytesCriticallyExhausted	Storage pool quota usage has crossed 90%.
CephPoolQuotaBytesNearExhaustion	Storage pool quota usage has crossed 70%.
OSDCPULoadHigh	CPU usage in the OSD container on a specific pod has exceeded 80%, potentially affecting the performance of the OSD.
PersistentVolumeUsageCritical	Persistent Volume Claim usage has exceeded more than 85% of its capacity.
PersistentVolumeUsageNearFull	Persistent Volume Claim usage has exceeded more than 75% of its capacity.

6.3.1. CephClusterCriticallyFull

Meaning	Storage cluster utilization has crossed 80% and will become read-only at 85%. Your Ceph cluster will become read-only once utilization crosses 85%. Free up some space or expand the storage cluster immediately. It is common to see alerts related to Object Storage Device (OSD) full or near full prior to this alert.
Impact	High

Diagnosis

Diagnosis

Scaling storage

Depending on the type of cluster, you need to add storage devices, nodes, or both. For more information, see the Scaling storage guide.

Mitigation

Deleting information

If it is not possible to scale up the cluster, you need to delete information to free up some space.

6.3.2. CephClusterErrorState

Meaning	This alert reflects that the storage cluster is in **ERROR** state for an unacceptable amount of time and thispts the storage availability. Check for other alerts that would have triggered prior to this one and troubleshoot those alerts first.
Impact	Critical

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
$ oc get pod | grep rook-ceph
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a rook-ceph that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

- If a node was assigned, check the kubelet on the node.
- If the basic health of the running pods, node affinity and resource availability on the nodes are verified, run the Ceph tools to get the status of the storage components.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.3. CephClusterNearFull

Meaning	Storage cluster utilization has crossed 75% and will become read-only at 85%. Free up some space or expand the storage cluster.
Impact	Critical

Diagnosis

Scaling storage

Depending on the type of cluster, you need to add storage devices, nodes, or both. For more information, see the Scaling storage guide.

Mitigation

Deleting information

If it is not possible to scale up the cluster, you need to delete information in order to free up some space.

6.3.4. CephClusterReadOnly

Meaning	Storage cluster utilization has crossed 85% and will become read-only now. Free up some space or expand the storage cluster immediately.
Impact	Critical

Diagnosis

Scaling storage

Depending on the type of cluster, you need to add storage devices, nodes, or both. For more information, see the Scaling storage guide.

Mitigation

Deleting information

If it is not possible to scale up the cluster, you need to delete information in order to free up some space.

6.3.5. CephClusterWarningState

Meaning	This alert reflects that the storage cluster has been in a warning state for an unacceptable amount of time. While the storage operations will continue to function in this state, it is recommended to fix the errors so that the cluster does not get into an error state. Check for other alerts that might have triggered prior to this one and troubleshoot those alerts first.
Impact	High

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep {ceph-component}
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.6. CephDataRecoveryTakingTooLong

Meaning	Data recovery is slow. Check whether all the Object Storage Devices (OSDs) are up and running.
Impact	High

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep rook-ceph-osd
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.7. CephMdsCacheUsageHigh

Meaning	When the storage metadata service (MDS) cannot keep its cache usage under the target threshold specified by **mds_health_cache_threshold**, or 150% of the cache limit set by **mds_cache_memory_limit**, the MDS sends a health alert to the monitors indicating the cache is too large. As a result, the MDS related operations become slow.
Impact	High

Diagnosis

The MDS tries to stay under a reservation of the **mds_cache_memory_limit** by trimming unused metadata in its cache and recalling cached items in the client caches. It is possible for the MDS to exceed this limit due to slow recall from clients as a result of multiple clients accesing the files.

Mitigation

Make sure you have enough memory provisioned for MDS cache. Memory resources for the MDS pods need to be updated in the **ocs-storageCluster** in order to increase the **mds_cache_memory_limit**. Run the following command to set the memory of MDS pods, for example, 16GB:

```
$ oc patch -n openshift-storage storagecluster ocs-storagecluster \
```

```
    --type merge \
    --patch '{"spec": {"resources": {"mds": {"limits": {"memory": "16Gi"},"requests": {"memory":
"16Gi"}}}}}'
```

OpenShift Data Foundation automatically sets **mds_cache_memory_limit** to half of the MDS pod memory limit. If the memory is set to 8GB using the previous command, then the operator sets the MDS cache memory limit to 4GB.

6.3.8. CephMdsCpuUsageHigh

Meaning	The storage metadata service (MDS) serves filesystem metadata. The MDS is crucial for any file creation, rename, deletion, and update operations. MDS by default is allocated two or three CPUs. This does not cause issues as long as there are not too many metadata operations. When the metadata operation load increases enough to trigger this alert, it means the default CPU allocation is unable to cope with load. You need to increase the CPU allocation.
Impact	High

Diagnosis

Click **Workloads → Pods**. Select the corresponding MDS pod and click on the **Metrics** tab. There you will see the allocated and used CPU. By default, the alert is fired if the used CPU is 67% of allocated CPU for 6 hours. If this is the case, follow the steps in the mitigation section.

Mitigation

You need to increase the allocated CPU.

Use the following command to set the number of allocated CPU for MDS, for example, **8**:

```
$ oc patch -n openshift-storage storagecluster ocs-storagecluster \
    --type merge \
    --patch '{"spec": {"resources": {"mds": {"limits": {"cpu": "8"},
    "requests": {"cpu": "8"}}}}}'
```

6.3.9. CephMdsMissingReplicas

Meaning	Minimum required replicas for the storage metadata service (MDS) are not available. MDS is responsible for filing metadata. Degradation of the MDS service can affect how the storage cluster works (related to the CephFS storage class) and should be fixed as soon as possible.
Impact	High

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep rook-ceph-mds
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.10. CephMgrIsAbsent

Meaning	Not having a Ceph manager running the monitoring of the cluster. Persistent Volume Claim (PVC) creation and deletion requests should be resolved as soon as possible.
Impact	High

Diagnosis

- Verify that the **rook-ceph-mgr** pod is failing, and restart if necessary. If the Ceph mgr pod restart fails, follow the general pod troubleshooting to resolve the issue.
  - Verify that the Ceph mgr pod is failing:

    ```
    $ oc get pods | grep mgr
    ```

  - Describe the Ceph mgr pod for more details:

    ```
    $ oc describe pods/<pod_name>
    ```

    <pod_name>
    Specify the **rook-ceph-mgr** pod name from the previous step.

    Analyze the errors related to resource issues.

  - Delete the pod, and wait for the pod to restart:

    ```
    $ oc get pods | grep mgr
    ```

Follow these steps for general pod troubleshooting:

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

   ```
   oc get pod | grep rook-ceph-mgr
   ```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

   ```
   # Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
   MYPOD=<pod_name>
   ```

   <pod_name>
   Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

   ```
   $ oc get pod/${MYPOD} -o wide
   ```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

  ```
  $ oc describe pod/${MYPOD}
  ```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.11. CephMgrIsMissingReplicas

Meaning	To resolve this alert, you need to determine the cause of the disappearance of the Ceph manager and restart if necessary.
Impact	High

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep rook-ceph-mgr
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

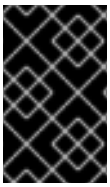
Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.12. CephMonHighNumberOfLeaderChanges

Meaning	In a Ceph cluster there is a redundant set of monitor pods that store critical information about the storage cluster. Monitor pods synchronize periodically to obtain information about the storage cluster. The first monitor pod to get the most updated information becomes the leader, and the other monitor pods will start their synchronization process after asking the leader. A problem in network connection or another kind of problem in one or more monitor pods produces an unusual change of the leader. This situation can negatively affect the storage cluster performance.
Impact	Medium



IMPORTANT

Check for any network issues. If there is a network issue, you need to escalate to the OpenShift Data Foundation team before you proceed with any of the following troubleshooting steps.

Diagnosis

1. Print the logs of the affected monitor pod to gather more information about the issue:

```
$ oc logs <rook-ceph-mon-X-yyyy> -n openshift-storage
```

<rook-ceph-mon-X-yyyy>

Specify the name of the affected monitor pod.

2. Alternatively, use the Openshift Web console to open the logs of the affected monitor pod. More information about possible causes is reflected in the log.

3. Perform the general pod troubleshooting steps:

pod status: pending

4. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep {ceph-component}
```

5. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

6. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.13. CephMonQuorumAtRisk

Meaning	Multiple MONs work together to provide redundancy. Each of the MONs keeps a copy of the metadata. The cluster is deployed with 3 MONs, and requires 2 or more MONs to be up and running for quorum and for the storage operations to run. If quorum is lost, access to data is at risk.
Impact	High

Diagnosis

Restore the Ceph MON Quorum. For more information, see *Restoring ceph-monitor quorum in OpenShift Data Foundation* in the Troubleshooting guide. If the restoration of the Ceph MON Quorum fails, follow the general pod troubleshooting to resolve the issue.

Perform the following for general pod troubleshooting:

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep rook-ceph-mon
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

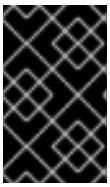
Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.14. CephMonQuorumLost

Meaning	In a Ceph cluster there is a redundant set of monitor pods that store critical information about the storage cluster. Monitor pods synchronize periodically to obtain information about the storage cluster. The first monitor pod to get the most updated information becomes the leader, and the other monitor pods will start their synchronization process after asking the leader. A problem in network connection or another kind of problem in one or more monitor pods produces an unusual change of the leader. This situation can negatively affect the storage cluster performance.
Impact	High



IMPORTANT

Check for any network issues. If there is a network issue, you need to escalate to the OpenShift Data Foundation team before you proceed with any of the following troubleshooting steps.

Diagnosis

Restore the Ceph MON Quorum. For more information, see *Restoring ceph-monitor quorum in OpenShift Data Foundation* in the Troubleshooting guide. If the restoration of the Ceph MON Quorum fails, follow the general pod troubleshooting to resolve the issue.

Alternatively, perform general pod troubleshooting:

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
oc get pod | grep {ceph-component}
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a {ceph-component}  that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

If a node was assigned, check the kubelet on the node.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.15. CephMonVersionMismatch

Meaning	Typically this alert triggers during an upgrade that is taking a long time.

Impact	Medium

Diagnosis

Check the **ocs-operator** subscription status and the operator pod health to check if an operator upgrade is in progress.

1. Check the **ocs-operator** subscription health.

```
$ oc get sub $(oc get pods -n openshift-storage | grep -v ocs-operator) -n openshift-storage -o json | jq .status.conditions
```

The status condition types are **CatalogSourcesUnhealthy**, **InstallPlanMissing**, **InstallPlanPending**, and **InstallPlanFailed**. The status for each type should be **False**.

Example output:

```
[
  {
    "lastTransitionTime": "2021-01-26T19:21:37Z",
    "message": "all available catalogsources are healthy",
    "reason": "AllCatalogSourcesHealthy",
    "status": "False",
    "type": "CatalogSourcesUnhealthy"
  }
]
```

The example output shows a **False** status for type **CatalogSourcesUnHealthly**, which means that the catalog sources are healthy.

2. Check the OCS operator pod status to see if there is an OCS operator upgrading in progress.

```
$ oc get pod -n openshift-storage | grep ocs-operator OCSOP=$(oc get pod -n openshift-storage -o custom-columns=POD:.metadata.name --no-headers | grep ocs-operator) echo $OCSOP oc get pod/${OCSOP} -n openshift-storage oc describe pod/${OCSOP} -n openshift-storage
```

If you determine that the `ocs-operator`is in progress, wait for 5 mins and this alert should resolve itself. If you have waited or see a different error status condition, continue troubleshooting.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

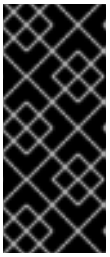
6.3.16. CephNodeDown

Meaning	A node running Ceph pods is down. While storage operations will continue to function as Ceph is designed to deal with a node failure, it is recommended to resolve the issue to minimize the risk of another node going down and affecting storage functions.
Impact	Medium

Diagnosis

1. List all the pods that are running and failing:

```
oc -n openshift-storage get pods
```



IMPORTANT

Ensure that you meet the OpenShift Data Foundation resource requirements so that the Object Storage Device (OSD) pods are scheduled on the new node. This may take a few minutes as the Ceph cluster recovers data for the failing but now recovering OSD. To watch this recovery in action, ensure that the OSD pods are correctly placed on the new worker node.

2. Check if the OSD pods that were previously failing are now running:

```
oc -n openshift-storage get pods
```

If the previously failing OSD pods have not been scheduled, use the **describe** command and check the events for reasons the pods were not rescheduled.

3. Describe the events for the failing OSD pod:

```
oc -n openshift-storage get pods | grep osd
```

4. Find the one or more failing OSD pods:

```
oc -n openshift-storage describe pods/<osd_podname_ from_the_ previous step>
```

In the events section look for the failure reasons, such as the resources are not being met.

In addition, you may use the **rook-ceph-toolbox** to watch the recovery. This step is optional, but is helpful for large Ceph clusters. To access the toolbox, run the following command:

```
TOOLS_POD=$(oc get pods -n openshift-storage -l app=rook-ceph-tools -o name)
oc rsh -n openshift-storage $TOOLS_POD
```

From the rsh command prompt, run the following, and watch for "recovery" under the io section:

```
ceph status
```

5. Determine if there are failed nodes.

   a. Get the list of worker nodes, and check for the node status:

```
oc get nodes --selector='node-role.kubernetes.io/worker','!node-role.kubernetes.io/infra'
```

b. Describe the node which is of the **NotReady** status to get more information about the failure:

```
oc describe node <node_name>
```

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.17. CephOSDCriticallyFull

Meaning	One of the Object Storage Devices (OSDs) is critically full. Expand the cluster immediately.
Impact	High

Diagnosis

Deleting data to free up storage space

You can delete data, and the cluster will resolve the alert through self healing processes.

IMPORTANT

This is only applicable to OpenShift Data Foundation clusters that are near or full but not in read-only mode. Read-only mode prevents any changes that include deleting data, that is, deletion of Persistent Volume Claim (PVC), Persistent Volume (PV) or both.

Expanding the storage capacity

Current storage size is less than 1 TB

You must first assess the ability to expand. For every 1 TB of storage added, the cluster needs to have 3 nodes each with a minimum available 2 vCPUs and 8 GiB memory.

You can increase the storage capacity to 4 TB via the add-on and the cluster will resolve the alert through self healing processes. If the minimum vCPU and memory resource requirements are not met, you need to add 3 additional worker nodes to the cluster.

Mitigation

- If your current storage size is equal to 4 TB, contact Red Hat support.
- Optional: Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.18. CephOSDDiskNotResponding

Meaning	A disk device is not responding. Check whether all the Object Storage Devices (OSDs) are up and running.
Impact	Medium

Diagnosis

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
$ oc get pod | grep rook-ceph
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a rook-ceph that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

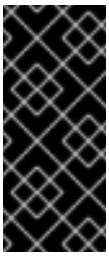
- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```



IMPORTANT

- If a node was assigned, check the kubelet on the node.
- If the basic health of the running pods, node affinity and resource availability on the nodes are verified, run the Ceph tools to get the status of the storage components.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.19. CephOSDDiskUnavailable

Meaning	A disk device is not accessible on one of the hosts and its corresponding Object Storage Device (OSD) is marked out by the Ceph cluster. This alert is raised when a Ceph node fails to recover within 10 minutes.
Impact	High

Diagnosis

Determine the failed node

1. Get the list of worker nodes, and check for the node status:

```
oc get nodes --selector='node-role.kubernetes.io/worker','!node-role.kubernetes.io/infra'
```

1. Describe the node which is of **NotReady** status to get more information on the failure:

```
oc describe node <node_name>
```

6.3.20. CephOSDFlapping

Meaning	A storage daemon has restarted 5 times in the last 5 minutes. Check the pod events or Ceph status to find out the cause.
Impact	High

Diagnosis

Follow the steps in the Flapping OSDs section of the Red Hat Ceph Storage Troubleshooting Guide.

Alternatively, follow the steps for general pod troubleshooting:

pod status: pending

1. Check for resource issues, pending Persistent Volume Claims (PVCs), node assignment, and kubelet problems:

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

```
$ oc get pod | grep rook-ceph
```

2. Set **MYPOD** as the variable for the pod that is identified as the problem pod:

```
# Examine the output for a rook-ceph that is in the pending state, not running or not ready
MYPOD=<pod_name>
```

<pod_name>

Specify the name of the pod that is identified as the problem pod.

3. Look for the resource limitations or pending PVCs. Otherwise, check for the node assignment:

```
$ oc get pod/${MYPOD} -o wide
```

pod status: NOT pending, running, but NOT ready

- Check the readiness probe:

```
$ oc describe pod/${MYPOD}
```

pod status: NOT pending, but NOT running

- Check for application or image issues:

```
$ oc logs pod/${MYPOD}
```

IMPORTANT

- If a node was assigned, check the kubelet on the node.
- If the basic health of the running pods, node affinity and resource availability on the nodes are verified, run the Ceph tools to get the status of the storage components.

Mitigation

Debugging log information

- This step is optional. Run the following command to gather the debugging information for the Ceph cluster:

```
$ oc adm must-gather --image=registry.redhat.io/odf4/odf-must-gather-rhel9:v4.15
```

6.3.21. CephOSDNearFull

Meaning	Utilization of back-end storage device Object Storage Device (OSD) has crossed 75% on a host.

Impact	High

Mitigation

Free up some space in the cluster, expand the storage cluster, or contact Red Hat support. For more information on scaling storage, see the Scaling storage guide.

6.3.22. CephOSDSlowOps

Meaning	An Object Storage Device (OSD) with slow requests is every OSD that is not able to service the I/O operations per second (IOPS) in the queue within the time defined by the **osd_op_complaint_time** parameter. By default, this parameter is set to 30 seconds.
Impact	Medium

Diagnosis

More information about the slow requests can be obtained using the Openshift console.

1. Access the OSD pod terminal, and run the following commands:

```
$ ceph daemon osd.<id> ops
```

```
$ ceph daemon osd.<id> dump_historic_ops
```

NOTE

The number of the OSD is seen in the pod name. For example, in **rook-ceph-osd-0-5d86d4d8d4-zlqkx**, **<0>** is the OSD.

Mitigation

The main causes of the OSDs having slow requests are:

- Problems with the underlying hardware or infrastructure, such as, disk drives, hosts, racks, or network switches. Use the Openshift monitoring console to find the alerts or errors about cluster resources. This can give you an idea about the root cause of the slow operations in the OSD.
- Problems with the network. These problems are usually connected with flapping OSDs. See the Flapping OSDs section of the Red Hat Ceph Storage Troubleshooting Guide
- If it is a network issue, escalate to the OpenShift Data Foundation team
- System load. Use the Openshift console to review the metrics of the OSD pod and the node which is running the OSD. Adding or assigning more resources can be a possible solution.

6.3.23. CephOSDVersionMismatch

Meaning	Typically this alert triggers during an upgrade that is taking a long time.
Impact	Medium

Diagnosis

Check the **ocs-operator** subscription status and the operator pod health to check if an operator upgrade is in progress.

1. Check the **ocs-operator** subscription health.

```
$ oc get sub $(oc get pods -n openshift-storage | grep -v ocs-operator) -n openshift-storage -o json | jq .status.conditions
```

The status condition types are **CatalogSourcesUnhealthy**, **InstallPlanMissing**, **InstallPlanPending**, and **InstallPlanFailed**. The status for each type should be **False**.

Example output:

```
[
  {
    "lastTransitionTime": "2021-01-26T19:21:37Z",
    "message": "all available catalogsources are healthy",
    "reason": "AllCatalogSourcesHealthy",
    "status": "False",
    "type": "CatalogSourcesUnhealthy"
  }
]
```

The example output shows a **False** status for type **CatalogSourcesUnHealthly**, which means that the catalog sources are healthy.

2. Check the OCS operator pod status to see if there is an OCS operator upgrading in progress.

```
$ oc get pod -n openshift-storage | grep ocs-operator OCSOP=$(oc get pod -n openshift-storage -o custom-columns=POD:.metadata.name --no-headers | grep ocs-operator) echo $OCSOP oc get pod/${OCSOP} -n openshift-storage oc describe pod/${OCSOP} -n openshift-storage
```

If you determine that the `ocs-operator`is in progress, wait for 5 mins and this alert should resolve itself. If you have waited or see a different error status condition, continue troubleshooting.

6.3.24. CephPGRepairTakingTooLong

Meaning	Self-healing operations are taking too long.
Impact	High

Diagnosis

Check for inconsistent Placement Groups (PGs), and repair them. For more information, see the Red Hat Knowledgebase solution Handle Inconsistent Placement Groups in Ceph .

6.3.25. CephPoolQuotaBytesCriticallyExhausted

Meaning	One or more pools has reached, or is very close to reaching, its quota. The threshold to trigger this error condition is controlled by the **mon_pool_quota_crit_threshold** configuration option.
Impact	High

Mitigation

Adjust the pool quotas. Run the following commands to fully remove or adjust the pool quotas up or down:

```
ceph osd pool set-quota <pool> max_bytes <bytes>
```

```
ceph osd pool set-quota <pool> max_objects <objects>
```

Setting the quota value to **0** will disable the quota.

6.3.26. CephPoolQuotaBytesNearExhaustion

Meaning	One or more pools is approaching a configured fullness threshold. One threshold that can trigger this warning condition is the **mon_pool_quota_warn_threshold** configuration option.
Impact	High

Mitigation

Adjust the pool quotas. Run the following commands to fully remove or adjust the pool quotas up or down:

```
ceph osd pool set-quota <pool> max_bytes <bytes>
```

```
ceph osd pool set-quota <pool> max_objects <objects>
```

Setting the quota value to **0** will disable the quota.

6.3.27. OSDCPULoadHigh

Meaning	OSD is a critical component in Ceph storage, responsible for managing data placement and recovery. High CPU usage in the OSD container suggests increased processing demands, potentially leading to degraded storage performance.
Impact	High

Diagnosis

1. Navigate to the Kubernetes dashboard or equivalent.
2. Access the **Workloads** section and select the relevant pod associated with the OSD alert.
3. Click the **Metrics** tab to view CPU metrics for the OSD container.
4. Verify that the CPU usage exceeds 80% over a significant period (as specified in the alert configuration).

Mitigation

If the OSD CPU usage is consistently high, consider taking the following steps:

1. Evaluate the overall storage cluster performance and identify the OSDs contributing to high CPU usage.
2. Increase the number of OSDs in the cluster by adding more new storage devices in the existing nodes or adding new nodes with new storage devices. Review the Scaling storage guide for instructions to help distribute the load and improve overall system performance.

6.3.28. PersistentVolumeUsageCritical

Meaning	A Persistent Volume Claim (PVC) is nearing its full capacity and may lead to data loss if not attended to timely.
Impact	High

Mitigation

Expand the PVC size to increase the capacity.

1. Log in to the OpenShift Web Console.
2. Click **Storage → PersistentVolumeClaim**.
3. Select **openshift-storage** from the **Project** drop-down list.
4. On the PVC you want to expand, click **Action menu (⋮) → Expand PVC**.
5. Update the **Total size** to the desired size.
6. Click **Expand**.

Alternatively, you can delete unnecessary data that may be taking up space.

6.3.29. PersistentVolumeUsageNearFull

Meaning	A Persistent Volume Claim (PVC) is nearing its full capacity and may lead to data loss if not attended to timely.
Impact	High

Mitigation

Expand the PVC size to increase the capacity.

1. Log in to the OpenShift Web Console.
2. Click **Storage → PersistentVolumeClaim**.
3. Select **openshift-storage** from the **Project** drop-down list.
4. On the PVC you want to expand, click **Action menu (⋮) → Expand PVC**.
5. Update the **Total size** to the desired size.
6. Click **Expand**.

Alternatively, you can delete unnecessary data that may be taking up space.

6.4. RESOLVING NOOBAA BUCKET ERROR STATE

Procedure

1. In the OpenShift Web Console, click **Storage → Data Foundation**.
2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.
3. Click the **Object** tab.
4. In the **Details** card, click the link under the **System Name** field.
5. In the left pane, click the **Buckets** option and search for the bucket in error state. If the bucket in error state is a namespace bucket, be sure to click the **Namespace Buckets** pane.
6. Click on its **Bucket Name**. Error encountered in bucket is displayed.
7. Depending on the specific error of the bucket, perform one or both of the following:
    a. For space related errors:
        i. In the left pane, click the **Resources** option.
        ii. Click on the resource in error state.
        iii. Scale the resource by adding more agents.
    b. For resource health errors:
        i. In the left pane, click the **Resources** option.
        ii. Click on the resource in error state.
        iii. Connectivity error means the backing service is not available and needs to be restored.
        iv. For access/permissions errors, update the connection's **Access Key** and **Secret Key**.

6.5. RESOLVING NOOBAA BUCKET EXCEEDING QUOTA STATE

To resolve **A NooBaa Bucket Is In Exceeding Quota State** error perform one of the following:

- Cleanup some of the data on the bucket.
- Increase the bucket quota by performing the following steps:
  1. In the OpenShift Web Console, click **Storage → Data Foundation**.
  2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.
  3. Click the **Object** tab.
  4. In the **Details** card, click the link under the **System Name** field.
  5. In the left pane, click the **Buckets** option and search for the bucket in error state.
  6. Click on its **Bucket Name**. Error encountered in bucket is displayed.
  7. Click **Bucket Policies → Edit Quota** and increase the quota.

6.6. RESOLVING NOOBAA BUCKET CAPACITY OR QUOTA STATE

Procedure

1. In the OpenShift Web Console, click **Storage → Data Foundation**.
2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.
3. Click the **Object** tab.
4. In the **Details** card, click the link under the **System Name** field.
5. In the left pane, click the **Resources** option and search for the PV pool resource.
6. For the PV pool resource with low capacity status, click on its **Resource Name**.
7. Edit the pool configuration and increase the number of agents.

6.7. RECOVERING PODS

When a first node (say **NODE1**) goes to NotReady state because of some issue, the hosted pods that are using PVC with ReadWriteOnce (RWO) access mode try to move to the second node (say **NODE2**) but get stuck due to multi-attach error. In such a case, you can recover MON, OSD, and application pods by using the following steps.

Procedure

1. Power off **NODE1** (from AWS or vSphere side) and ensure that **NODE1** is completely down.
2. Force delete the pods on **NODE1** by using the following command:

```
$ oc delete pod <pod-name> --grace-period=0 --force
```

6.8. RECOVERING FROM EBS VOLUME DETACH

When an OSD or MON elastic block storage (EBS) volume where the OSD disk resides is detached from the worker Amazon EC2 instance, the volume gets reattached automatically within one or two minutes. However, the OSD pod gets into a **CrashLoopBackOff** state. To recover and bring back the pod to **Running** state, you must restart the EC2 instance.

6.9. ENABLING AND DISABLING DEBUG LOGS FOR ROOK-CEPH-OPERATOR

Enable the debug logs for the rook-ceph-operator to obtain information about failures that help in troubleshooting issues.

Procedure

Enabling the debug logs

1. Edit the configmap of the rook-ceph-operator.

```
$ oc edit configmap rook-ceph-operator-config
```

2. Add the **ROOK_LOG_LEVEL: DEBUG** parameter in the **rook-ceph-operator-config** yaml file to enable the debug logs for rook-ceph-operator.

```
…
data:
  # The logging level for the operator: INFO | DEBUG
  ROOK_LOG_LEVEL: DEBUG
```

Now, the rook-ceph-operator logs consist of the debug information.

Disabling the debug logs

1. Edit the configmap of the rook-ceph-operator.

```
$ oc edit configmap rook-ceph-operator-config
```

2. Add the **ROOK_LOG_LEVEL: INFO** parameter in the **rook-ceph-operator-config** yaml file to disable the debug logs for rook-ceph-operator.

```
…
data:
  # The logging level for the operator: INFO | DEBUG
  ROOK_LOG_LEVEL: INFO
```

6.10. RESOLVING LOW CEPH MONITOR COUNT ALERT

The **CephMonLowNumber** alert is displayed in the notification panel or Alert Center of the OpenShift Web Console to indicate the low number of Ceph monitor count when your internal mode deployment has five or more nodes, racks, or rooms, and when there are five or more number of failure domains in the deployment. You can increase the Ceph monitor count to improve the availability of cluster.

Procedure

1. In the **CephMonLowNumber** alert of the notification panel or Alert Center of OpenShift Web Console, click **Configure**.

2. In the **Configure Ceph Monitor** pop up, click **Update** count.
   In the pop up, the recommended monitor count depending on the number of failure zones is shown.

3. In the **Configure CephMon** pop up, update the monitor count value based on the recommended value and click **Save changes**.

6.11. TROUBLESHOOTING UNHEALTHY BLOCKLISTED NODES

6.11.1. ODFRBDClientBlocked

Meaning	This alert indicates that an RADOS Block Device (RBD) client might be blocked by Ceph on a specific node within your Kubernetes cluster. The blocklisting occurs when the **ocs_rbd_client_blocklisted metric** reports a value of 1 for the node. Additionally, there are pods in a **CreateContainerError** state on the same node. The blocklisting can potentially result in the filesystem for the Persistent Volume Claims (PVCs) using RBD becoming read-only. It is crucial to investigate this alert to prevent any disruption to your storage cluster.
Impact	High

Diagnosis

The blocklisting of an RBD client can occur due to several factors, such as network or cluster slowness. In certain cases, the exclusive lock contention among three contending clients (workload, mirror daemon, and manager/scheduler) can lead to the blocklist.

Mitigation

1. Taint the blocklisted node: In Kubernetes, consider tainting the node that is blocklisted to trigger the eviction of pods to another node. This approach relies on the assumption that the unmounting/unmapping process progresses gracefully. Once the pods have been successfully evicted, the blocklisted node can be untainted, allowing the blocklist to be cleared. The pods can then be moved back to the untainted node.

2. Reboot the blocklisted node: If tainting the node and evicting the pods do not resolve the blocklisting issue, a reboot of the blocklisted node can be attempted. This step may help alleviate any underlying issues causing the blocklist and restore normal functionality.

IMPORTANT

Investigating and resolving the blocklist issue promptly is essential to avoid any further impact on the storage cluster.

CHAPTER 7. CHECKING FOR LOCAL STORAGE OPERATOR DEPLOYMENTS

Red Hat OpenShift Data Foundation clusters with Local Storage Operator are deployed using local storage devices. To find out if your existing cluster with OpenShift Data Foundation was deployed using local storage devices, use the following procedure:

Prerequisites

- OpenShift Data Foundation is installed and running in the **openshift-storage** namespace.

Procedure

By checking the storage class associated with your OpenShift Data Foundation cluster's persistent volume claims (PVCs), you can tell if your cluster was deployed using local storage devices.

1. Check the storage class associated with OpenShift Data Foundation cluster's PVCs with the following command:

```
$ oc get pvc -n openshift-storage
```

2. Check the output. For clusters with Local Storage Operators, the PVCs associated with **ocs-deviceset** use the storage class **localblock**. The output looks similar to the following:

```
NAME                  STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS              AGE
db-noobaa-db-0        Bound    pvc-d96c747b-2ab5-47e2-b07e-1079623748d8   50Gi
RWO          ocs-storagecluster-ceph-rbd   114s
ocs-deviceset-0-0-lzfrd   Bound    local-pv-7e70c77c                       1769Gi     RWO
localblock                2m10s
ocs-deviceset-1-0-7rggl   Bound    local-pv-b19b3d48                       1769Gi     RWO
localblock                2m10s
ocs-deviceset-2-0-znhk8   Bound    local-pv-e9f22cdc                       1769Gi     RWO
localblock                2m10s
```

Additional Resources

- Deploying OpenShift Data Foundation using local storage devices on VMware
- Deploying OpenShift Data Foundation using local storage devices on Red Hat Virtualization
- Deploying OpenShift Data Foundation using local storage devices on bare metal
- Deploying OpenShift Data Foundation using local storage devices on IBM Power

CHAPTER 8. REMOVING FAILED OR UNWANTED CEPH OBJECT STORAGE DEVICES

The failed or unwanted Ceph OSDs (Object Storage Devices) affects the performance of the storage infrastructure. Hence, to improve the reliability and resilience of the storage cluster, you must remove the failed or unwanted Ceph OSDs.

If you have any failed or unwanted Ceph OSDs to remove:

1. Verify the Ceph health status.
   For more information see: Verifying Ceph cluster is healthy.
2. Based on the provisioning of the OSDs, remove failed or unwanted Ceph OSDs.
   See:
   - Removing failed or unwanted Ceph OSDs in dynamically provisioned Red Hat OpenShift Data Foundation.
   - Removing failed or unwanted Ceph OSDs provisioned using local storage devices .

If you are using local disks, you can reuse these disks after removing the old OSDs.

8.1. VERIFYING CEPH CLUSTER IS HEALTHY

Storage health is visible on the **Block** and **File** and **Object** dashboards.

Procedure

1. In the OpenShift Web Console, click **Storage → Data Foundation**.
2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.
3. In the **Status** card of the **Block and File** tab, verify that the *Storage Cluster* has a green tick.
4. In the **Details** card, verify that the cluster information is displayed.

8.2. REMOVING FAILED OR UNWANTED CEPH OSDS IN DYNAMICALLY PROVISIONED RED HAT OPENSHIFT DATA FOUNDATION

Follow the steps in the procedure to remove the failed or unwanted Ceph Object Storage Devices (OSDs) in dynamically provisioned Red Hat OpenShift Data Foundation.

IMPORTANT

Scaling down of clusters is supported only with the help of the Red Hat support team.

WARNING

- Removing an OSD when the Ceph component is not in a healthy state can result in data loss.
- Removing two or more OSDs at the same time results in data loss.

Prerequisites

- Check if Ceph is healthy. For more information see Verifying Ceph cluster is healthy.
- Ensure no alerts are firing or any rebuilding process is in progress.

Procedure

1. Scale down the OSD deployment.

```
# oc scale deployment rook-ceph-osd-<osd-id> --replicas=0
```

2. Get the **osd-prepare** pod for the Ceph OSD to be removed.

```
# oc get deployment rook-ceph-osd-<osd-id> -oyaml | grep ceph.rook.io/pvc
```

3. Delete the **osd-prepare** pod.

```
# oc delete -n openshift-storage pod rook-ceph-osd-prepare-<pvc-from-above-command>-<pod-suffix>
```

4. Remove the failed OSD from the cluster.

```
# failed_osd_id=<osd-id>

# oc process -n openshift-storage ocs-osd-removal -p FAILED_OSD_IDS=$<failed_osd_id> | oc create -f -
```

where, **FAILED_OSD_ID** is the integer in the pod name immediately after the **rook-ceph-osd** prefix.

5. Verify that the OSD is removed successfully by checking the logs.

```
# oc logs -n openshift-storage ocs-osd-removal-$<failed_osd_id>-<pod-suffix>
```

6. Optional: If you get an error as **cephosd:osd.0 is NOT ok to destroy** from the **ocs-osd-removal-job** pod in OpenShift Container Platform, see Troubleshooting the error **cephosd:osd.0 is NOT ok to destroy** while removing failed or unwanted Ceph OSDs .

7. Delete the OSD deployment.

```
# oc delete deployment rook-ceph-osd-<osd-id>
```

Verification step

- To check if the OSD is deleted successfully, run:

```
# oc get pod -n openshift-storage ocs-osd-removal-$<failed_osd_id>-<pod-suffix>
```

This command must return the status as **Completed**.

8.3. REMOVING FAILED OR UNWANTED CEPH OSDS PROVISIONED USING LOCAL STORAGE DEVICES

You can remove failed or unwanted Ceph provisioned Object Storage Devices (OSDs) using local storage devices by following the steps in the procedure.

IMPORTANT

Scaling down of clusters is supported only with the help of the Red Hat support team.

WARNING

- Removing an OSD when the Ceph component is not in a healthy state can result in data loss.
- Removing two or more OSDs at the same time results in data loss.

Prerequisites

- Check if Ceph is healthy. For more information see Verifying Ceph cluster is healthy.
- Ensure no alerts are firing or any rebuilding process is in progress.

Procedure

1. Forcibly, mark the OSD down by scaling the replicas on the OSD deployment to 0. You can skip this step if the OSD is already down due to failure.

```
# oc scale deployment rook-ceph-osd-<osd-id> --replicas=0
```

2. Remove the failed OSD from the cluster.

```
# failed_osd_id=<osd_id>

# oc process -n openshift-storage ocs-osd-removal -p FAILED_OSD_IDS=$<failed_osd_id> |
oc create -f -
```

where, **FAILED_OSD_ID** is the integer in the pod name immediately after the **rook-ceph-osd** prefix.

3. Verify that the OSD is removed successfully by checking the logs.

```
# oc logs -n openshift-storage ocs-osd-removal-$<failed_osd_id>-<pod-suffix>
```

4. Optional: If you get an error as **cephosd:osd.0 is NOT ok to destroy** from the **ocs-osd-removal-job** pod in OpenShift Container Platform, see Troubleshooting the error **cephosd:osd.0 is NOT ok to destroy** while removing failed or unwanted Ceph OSDs .

5. Delete persistent volume claim (PVC) resources associated with the failed OSD.

   a. Get the **PVC** associated with the failed OSD.

   ```
   # oc get -n openshift-storage -o yaml deployment rook-ceph-osd-<osd-id> | grep ceph.rook.io/pvc
   ```

   b. Get the **persistent volume** (PV) associated with the PVC.

   ```
   # oc get -n openshift-storage pvc <pvc-name>
   ```

   c. Get the failed device name.

   ```
   # oc get pv <pv-name-from-above-command> -oyaml | grep path
   ```

   d. Get the **prepare-pod** associated with the failed OSD.

   ```
   # oc describe -n openshift-storage pvc ocs-deviceset-0-0-nvs68 | grep Mounted
   ```

   e. Delete the **osd-prepare pod** before removing the associated PVC.

   ```
   # oc delete -n openshift-storage pod <osd-prepare-pod-from-above-command>
   ```

   f. Delete the **PVC** associated with the failed OSD.

   ```
   # oc delete -n openshift-storage pvc <pvc-name-from-step-a>
   ```

6. Remove failed device entry from the **LocalVolume custom resource** (CR).

   a. Log in to node with the failed device.

   ```
   # oc debug node/<node_with_failed_osd>
   ```

   b. Record the /dev/disk/by-id/<id> for the failed device name.

   ```
   # ls -alh /mnt/local-storage/localblock/
   ```

7. Optional: In case, Local Storage Operator is used for provisioning OSD, login to the machine with {osd-id} and remove the device symlink.

   ```
   # oc debug node/<node_with_failed_osd>
   ```

   a. Get the OSD symlink for the failed device name.

   ```
   # ls -alh /mnt/local-storage/localblock
   ```

b. Remove the symlink.

```
# rm /mnt/local-storage/localblock/<failed-device-name>
```

8. Delete the PV associated with the OSD.

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

Verification step

- To check if the OSD is deleted successfully, run:

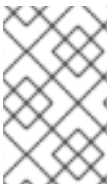
```
#oc get pod -n openshift-storage ocs-osd-removal-$<failed_osd_id>-<pod-suffix>
```

This command must return the status as **Completed**.

8.4. TROUBLESHOOTING THE ERROR CEPHOSD:OSD.0 IS NOT OK TO DESTROY WHILE REMOVING FAILED OR UNWANTED CEPH OSDS

If you get an error as **cephosd:osd.0 is NOT ok to destroy** from the **ocs-osd-removal-job** pod in OpenShift Container Platform, run the Object Storage Device (OSD) removal job with **FORCE_OSD_REMOVAL** option to move the OSD to a destroyed state.

```
# oc process -n openshift-storage ocs-osd-removal -p FORCE_OSD_REMOVAL=true -p
FAILED_OSD_IDS=$<failed_osd_id> | oc create -f -
```



NOTE

You must use the **FORCE_OSD_REMOVAL** option only if all the PGs are in active state. If not, PGs must either complete the back filling or further investigate to ensure they are active.

CHAPTER 9. TROUBLESHOOTING AND DELETING REMAINING RESOURCES DURING UNINSTALL

Occasionally some of the custom resources managed by an operator may remain in "Terminating" status waiting on the finalizer to complete, although you have performed all the required cleanup tasks. In such an event you need to force the removal of such resources. If you do not do so, the resources remain in the **Terminating** state even after you have performed all the uninstall steps.

1. Check if the openshift-storage namespace is stuck in the **Terminating** state upon deletion.

```
$ oc get project -n <namespace>
```

Output:

```
NAME                DISPLAY NAME   STATUS
openshift-storage                  Terminating
```

2. Check for the **NamespaceFinalizersRemaining** and **NamespaceContentRemaining** messages in the **STATUS** section of the command output and perform the next step for each of the listed resources.

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

Example output :

```
status:
  conditions:
  - lastTransitionTime: "2020-07-26T12:32:56Z"
    message: All resources successfully discovered
    reason: ResourcesDiscovered
    status: "False"
    type: NamespaceDeletionDiscoveryFailure
  - lastTransitionTime: "2020-07-26T12:32:56Z"
    message: All legacy kube types successfully parsed
    reason: ParsedGroupVersions
    status: "False"
    type: NamespaceDeletionGroupVersionParsingFailure
  - lastTransitionTime: "2020-07-26T12:32:56Z"
    message: All content successfully deleted, may be waiting on finalization
    reason: ContentDeleted
    status: "False"
    type: NamespaceDeletionContentFailure
  - lastTransitionTime: "2020-07-26T12:32:56Z"
    message: 'Some resources are remaining: cephobjectstoreusers.ceph.rook.io has
      1 resource instances'
    reason: SomeResourcesRemain
    status: "True"
    type: NamespaceContentRemaining
  - lastTransitionTime: "2020-07-26T12:32:56Z"
    message: 'Some content in the namespace has finalizers remaining:
cephobjectstoreuser.ceph.rook.io
      in 1 resource instances'
```

```
    reason: SomeFinalizersRemain
    status: "True"
    type: NamespaceFinalizersRemaining
```

3. Delete all the remaining resources listed in the previous step.
   For each of the resources to be deleted, do the following:

   a. Get the object kind of the resource which needs to be removed. See the message in the above output.
      Example :

      message: Some content in the namespace has finalizers remaining: cephobjectstoreuser.ceph.rook.io

      Here **cephobjectstoreuser.ceph.rook.io** is the object kind.

   b. Get the Object name corresponding to the object kind.

      ```
      $ oc get  <Object-kind> -n  <project-name>
      ```

      Example :

      ```
      $ oc get cephobjectstoreusers.ceph.rook.io -n openshift-storage
      ```

      Example output:

      ```
      NAME                          AGE
      noobaa-ceph-objectstore-user   26h
      ```

   c. Patch the resources.

      ```
      $ oc patch -n <project-name> <object-kind>/<object-name> --type=merge -p
      '{"metadata": {"finalizers":null}}'
      ```

      Example:

      ```
      $ oc patch -n openshift-storage cephobjectstoreusers.ceph.rook.io/noobaa-ceph-
      objectstore-user \
      --type=merge -p '{"metadata": {"finalizers":null}}'
      ```

      Output:

      ```
      cephobjectstoreuser.ceph.rook.io/noobaa-ceph-objectstore-user patched
      ```

4. Verify that the openshift-storage project is deleted.

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

   Output:

   ```
   Error from server (NotFound): namespaces "openshift-storage" not found
   ```

   If the issue persists, reach out to Red Hat Support.

CHAPTER 10. TROUBLESHOOTING CEPHFS PVC CREATION IN EXTERNAL MODE

If you have updated the Red Hat Ceph Storage cluster from a version lower than 4.1.1 to the latest release and is not a freshly deployed cluster, you must manually set the application type for the CephFS pool on the Red Hat Ceph Storage cluster to enable CephFS Persistent Volume Claim (PVC) creation in external mode.

1. Check for CephFS pvc stuck in **Pending** status.

```
# oc get pvc -n <namespace>
```

Example output :

```
NAME                     STATUS    VOLUME
CAPACITY  ACCESS MODES    STORAGECLASS                  AGE
ngx-fs-pxknkcix20-pod     Pending
                    ocs-external-storagecluster-cephfs  28h
[...]
```

2. Check the output of the **oc describe** command to see the events for respective pvc. Expected error message is **cephfs_metadata/csi.volumes.default/csi.volume.pvc-xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx: (1) Operation not permitted)**

```
# oc describe pvc ngx-fs-pxknkcix20-pod -n nginx-file
```

Example output:

```
Name:          ngx-fs-pxknkcix20-pod
Namespace:     nginx-file
StorageClass:  ocs-external-storagecluster-cephfs
Status:        Pending
Volume:
Labels:        <none>
Annotations:   volume.beta.kubernetes.io/storage-provisioner: openshift-
storage.cephfs.csi.ceph.com
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    ngx-fs-oyoe047v2bn2ka42jfgg-pod-hqhzf
Events:
 Type     Reason              Age                 From
Message
 ----     ------              ----                ----
-------
 Warning  ProvisioningFailed  107m (x245 over 22h)  openshift-
storage.cephfs.csi.ceph.com_csi-cephfsplugin-provisioner-5f8b66cc96-hvcqp_6b7044af-
c904-4795-9ce5-bf0cf63cc4a4
 (combined from similar events): failed to provision volume with StorageClass "ocs-external-
storagecluster-cephfs": rpc error: code = Internal desc = error (an error (exit status 1)
occurred while
 running rados args: [-m 192.168.13.212:6789,192.168.13.211:6789,192.168.13.213:6789 --
id csi-cephfs-provisioner --keyfile=stripped -c /etc/ceph/ceph.conf -p cephfs_metadata
```

```
getomapval
 csi.volumes.default csi.volume.pvc-1ac0c6e6-9428-445d-bbd6-1284d54ddb47 /tmp/omap-get-186436239 --namespace=csi]) occurred, command output streams is ( error getting omap value
 cephfs_metadata/csi.volumes.default/csi.volume.pvc-1ac0c6e6-9428-445d-bbd6-1284d54ddb47: (1) Operation not permitted)
```

3. Check the settings for the **<cephfs metadata pool name>** (here **cephfs_metadata**) and **<cephfs data pool name>** (here **cephfs_data**). For running the command, you will need **jq** preinstalled in the Red Hat Ceph Storage client node.

```
# ceph osd pool ls detail --format=json | jq '.[] | select(.pool_name| startswith("cephfs")) | .pool_name, .application_metadata' "cephfs_data"
{
  "cephfs": {}
}
"cephfs_metadata"
{
   "cephfs": {}
}
```

4. Set the application type for the CephFS pool.

   - Run the following commands on the Red Hat Ceph Storage client node :

     ```
     # ceph osd pool application set <cephfs metadata pool name> cephfs metadata cephfs
     ```

     ```
     # ceph osd pool application set <cephfs data pool name> cephfs data cephfs
     ```

5. Verify if the settings are applied.

```
 # ceph osd pool ls detail --format=json | jq '.[] | select(.pool_name| startswith("cephfs")) | .pool_name, .application_metadata' "cephfs_data"
 {
   "cephfs": {
     "data": "cephfs"
  }
 }
 "cephfs_metadata"
 {
   "cephfs": {
     "metadata": "cephfs"
   }
 }
```

6. Check the CephFS PVC status again. The PVC should now be in **Bound** state.

```
# oc get pvc -n <namespace>
```

Example output :

```
NAME                     STATUS    VOLUME
CAPACITY  ACCESS MODES    STORAGECLASS                  AGE
ngx-fs-pxknkcix20-pod    Bound     pvc-1ac0c6e6-9428-445d-bbd6-1284d54ddb47
```

```
1Mi        RWO            ocs-external-storagecluster-cephfs  29h
[...]
```

CHAPTER 11. RESTORING THE MONITOR PODS IN OPENSHIFT DATA FOUNDATION

Restore the monitor pods if all three of them go down, and when OpenShift Data Foundation is not able to recover the monitor pods automatically.

NOTE

This is a disaster recovery procedure and must be performed under the guidance of the Red Hat support team. Contact Red Hat support team on, Red Hat support.

Procedure

1. Scale down the **rook-ceph-operator** and **ocs operator** deployments.

```
# oc scale deployment rook-ceph-operator --replicas=0 -n openshift-storage
```

```
# oc scale deployment ocs-operator --replicas=0 -n openshift-storage
```

2. Create a backup of all deployments in **openshift-storage** namespace.

```
# mkdir backup
```

```
# cd backup
```

```
# oc project openshift-storage
```

```
# for d in $(oc get deployment|awk -F' ' '{print $1}'|grep -v NAME); do echo $d;oc get deployment $d -o yaml > oc_get_deployment.${d}.yaml; done
```

3. Patch the Object Storage Device (OSD) deployments to remove the **livenessProbe** parameter, and run it with the command parameter as **sleep**.

```
# for i in $(oc get deployment -l app=rook-ceph-osd -oname);do oc patch ${i} -n openshift-storage --type='json' -p '[{"op":"remove", "path":"/spec/template/spec/containers/0/livenessProbe"}]' ; oc patch ${i} -n openshift-storage -p '{"spec": {"template": {"spec": {"containers": [{"name": "osd", "command": ["sleep", "infinity"], "args": []}]}}}}' ; done
```

4. Retrieve the **monstore** cluster map from all the OSDs.

   a. Create the **recover_mon.sh** script.

```
#!/bin/bash
ms=/tmp/monstore

rm -rf $ms
mkdir $ms

for osd_pod in $(oc get po -l app=rook-ceph-osd -oname -n openshift-storage); do
```

```
  echo "Starting with pod: $osd_pod"

  podname=$(echo $osd_pod|sed 's/pod\///g')
  oc exec $osd_pod -- rm -rf $ms
  oc cp $ms $podname:$ms

  rm -rf $ms
  mkdir $ms

  echo "pod in loop: $osd_pod ; done deleting local dirs"

  oc exec $osd_pod -- ceph-objectstore-tool --type bluestore --data-path
/var/lib/ceph/osd/ceph-$(oc get $osd_pod -ojsonpath='{
.metadata.labels.ceph_daemon_id }') --op update-mon-db --no-mon-config --mon-store-
path $ms
  echo "Done with COT on pod: $osd_pod"

  oc cp $podname:$ms $ms

  echo "Finished pulling COT data from pod: $osd_pod"
done
```

b. Run the **recover_mon.sh** script.

```
# chmod +x recover_mon.sh
```

```
# ./recover_mon.sh
```

5. Patch the MON deployments, and run it with the command parameter as **sleep**.

a. Edit the MON deployments.

```
# for i in $(oc get deployment -l app=rook-ceph-mon -oname);do oc patch ${i} -n
openshift-storage -p '{"spec": {"template": {"spec": {"containers": [{"name": "mon",
"command": ["sleep", "infinity"], "args": []}]}}}}'; done
```

b. Patch the MON deployments to increase the **initialDelaySeconds**.

```
# oc get deployment rook-ceph-mon-a -o yaml | sed "s/initialDelaySeconds:
10/initialDelaySeconds: 2000/g" | oc replace -f -
```

```
# oc get deployment rook-ceph-mon-b -o yaml | sed "s/initialDelaySeconds:
10/initialDelaySeconds: 2000/g" | oc replace -f -
```

```
# oc get deployment rook-ceph-mon-c -o yaml | sed "s/initialDelaySeconds:
10/initialDelaySeconds: 2000/g" | oc replace -f -
```

6. Copy the previously retrieved **monstore** to the **mon-a** pod.

```
# oc cp /tmp/monstore/ $(oc get po -l app=rook-ceph-mon,mon=a -oname |sed
's/pod\///g'):/tmp/
```

7. Navigate into the MON pod and change the ownership of the retrieved **monstore**.

```
# oc rsh $(oc get po -l app=rook-ceph-mon,mon=a -oname)
```

```
# chown -R ceph:ceph /tmp/monstore
```

8. Copy the keyring template file before rebuilding the **mon db**.

```
# oc rsh $(oc get po -l app=rook-ceph-mon,mon=a -oname)
```

```
# cp /etc/ceph/keyring-store/keyring /tmp/keyring
```

```
# cat /tmp/keyring
 [mon.]
   key = AQCleqldWqm5IhAAgZQbEzoShkZV42RiQVffnA==
   caps mon = "allow *"
 [client.admin]
   key = AQCmAKld8J05KxAArOWeRAw63gAwwZO5o75ZNQ==
   auid = 0
   caps mds = "allow *"
   caps mgr = "allow *"
   caps mon = "allow *"
   caps osd = "allow *”
```

9. Identify the keyring of all other Ceph daemons (MGR, MDS, RGW, Crash, CSI and CSI provisioners) from its respective secrets.

```
# oc get secret rook-ceph-mds-ocs-storagecluster-cephfilesystem-a-keyring -ojson  | jq
.data.keyring | xargs echo | base64 -d

[mds.ocs-storagecluster-cephfilesystem-a]
key = AQB3r8VgAtr6OhAAVhhXpNKqRTuEVdRoxG4uRA==
caps mon = "allow profile mds"
caps osd = "allow *"
caps mds = "allow"
```

Example keyring file, **/etc/ceph/ceph.client.admin.keyring**:

```
[mon.]
 key = AQDxTF1hNgLTNxAAi51cCojs01b4I5E6v2H8Uw==
 caps mon = "allow "
[client.admin]
        key = AQDxTF1hpzguOxAA0sS8nN4udoO35OEbt3bqMQ==
        caps mds = "allow " caps mgr = "allow *" caps mon = "allow *" caps osd = "allow *"
[mds.ocs-storagecluster-cephfilesystem-a] key =
AQCKTV1horgjARAA8aF/BDh/4+eG4RCNBCl+aw== caps mds = "allow" caps mon = "allow
profile mds" caps osd = "allow *" [mds.ocs-storagecluster-cephfilesystem-b] key =
AQCKTV1hN4gKLBAA5emIVq3ncV7AMEM1c1RmGA== caps mds = "allow" caps mon =
"allow profile mds" caps osd = "allow *" [client.rgw.ocs.storagecluster.cephobjectstore.a] key
= AQCOkdBixmpiAxAA4X7zjn6SGTI9c1MBflszYA== caps mon = "allow rw" caps osd =
"allow rwx" [mgr.a] key = AQBOTV1hGYOEORAA87471+eIZLZtptfkcHvTRg== caps mds =
"allow *" caps mon = "allow profile mgr" caps osd = "allow *" [client.crash] key =
AQBOTV1htO1aGRAAe2MPYcGdiAT+Oo4CNPSF1g== caps mgr = "allow rw" caps mon =
"allow profile crash" [client.csi-cephfs-node] key =
AQBOTV1hiAtuBBAAaPPBVgh1AqZJlDeHWdoFLw== caps mds = "allow rw" caps mgr =
```

```
"allow rw" caps mon = "allow r" caps osd = "allow rw tag cephfs *=" [client.csi-cephfs-
provisioner] key = AQBNTV1hHu6wMBAAzNXZv36aZJuE1iz7S7GfeQ== caps mgr = "allow
rw" caps mon = "allow r" caps osd = "allow rw tag cephfs metadata="
[client.csi-rbd-node]
 key = AQBNTV1h+LnkIRAAWnpIN9bUAmSHOvJ0EJXHRw==
 caps mgr = "allow rw"
 caps mon = "profile rbd"
 caps osd = "profile rbd"
[client.csi-rbd-provisioner]
 key = AQBNTV1hMNcsExAAvA3gHB2qaY33LOdWCvHG/A==
 caps mgr = "allow rw"
 caps mon = "profile rbd"
 caps osd = "profile rbd"
```

IMPORTANT

- For **client.csi** related keyring, refer to the previous keyring file output and add the default **caps** after fetching the key from its respective OpenShift Data Foundation secret.
- OSD keyring is added automatically post recovery.

10. Navigate into the **mon-a** pod, and verify that the **monstore** has a **monmap**.

    a. Navigate into the **mon-a** pod.

    ```
    # oc rsh $(oc get po -l app=rook-ceph-mon,mon=a -oname)
    ```

    b. Verify that the **monstore** has a **monmap**.

    ```
    # ceph-monstore-tool /tmp/monstore get monmap -- --out /tmp/monmap
    ```

    ```
    # monmaptool /tmp/monmap --print
    ```

11. Optional: If the **monmap** is missing then create a new **monmap**.

    ```
    # monmaptool --create --add <mon-a-id> <mon-a-ip> --add <mon-b-id> <mon-b-ip> --add
    <mon-c-id> <mon-c-ip> --enable-all-features --clobber /root/monmap --fsid <fsid>
    ```

    <mon-a-id>

    Is the ID of the **mon-a** pod.

    <mon-a-ip>

    Is the IP address of the **mon-a** pod.

    <mon-b-id>

    Is the ID of the **mon-b** pod.

    <mon-b-ip>

    Is the IP address of the **mon-b** pod.

    <mon-c-id>

    Is the ID of the **mon-c** pod.

    <mon-c-ip>

Is the IP address of the **mon-c** pod.

<fsid>

Is the file system ID.

12. Verify the **monmap**.

```
# monmaptool /root/monmap --print
```

13. Import the **monmap**.

IMPORTANT

Use the previously created **keyring** file.

```
# ceph-monstore-tool /tmp/monstore rebuild -- --keyring /tmp/keyring --monmap
/root/monmap
```

```
# chown -R ceph:ceph /tmp/monstore
```

14. Create a backup of the old **store.db** file.

```
# mv /var/lib/ceph/mon/ceph-a/store.db /var/lib/ceph/mon/ceph-a/store.db.corrupted
```

```
# mv /var/lib/ceph/mon/ceph-b/store.db /var/lib/ceph/mon/ceph-b/store.db.corrupted
```

```
# mv /var/lib/ceph/mon/ceph-c/store.db /var/lib/ceph/mon/ceph-c/store.db.corrupted
```

15. Copy the rebuild **store.db** file to the **monstore** directory.

```
# mv /tmp/monstore/store.db /var/lib/ceph/mon/ceph-a/store.db
```

```
# chown -R ceph:ceph /var/lib/ceph/mon/ceph-a/store.db
```

16. After rebuilding the **monstore** directory, copy the **store.db** file from local to the rest of the MON pods.

```
# oc cp $(oc get po -l app=rook-ceph-mon,mon=a -oname | sed
's/pod\///g'):/var/lib/ceph/mon/ceph-a/store.db /tmp/store.db
```

```
# oc cp /tmp/store.db $(oc get po -l app=rook-ceph-mon,mon=<id> -oname | sed
's/pod\///g'):/var/lib/ceph/mon/ceph-<id>
```

<id>

Is the ID of the MON pod

17. Navigate into the rest of the MON pods and change the ownership of the copied **monstore**.

```
# oc rsh $(oc get po -l app=rook-ceph-mon,mon=<id> -oname)
```

```
# chown -R ceph:ceph /var/lib/ceph/mon/ceph-<id>/store.db
```

<id>

Is the ID of the MON pod

18. Revert the patched changes.

- For MON deployments:

```
# oc replace --force -f <mon-deployment.yaml>
```

<mon-deployment.yaml>

Is the MON deployment yaml file

- For OSD deployments:

```
# oc replace --force -f <osd-deployment.yaml>
```

<osd-deployment.yaml>

Is the OSD deployment yaml file

- For MGR deployments:

```
# oc replace --force -f <mgr-deployment.yaml>
```

<mgr-deployment.yaml>

Is the MGR deployment yaml file

IMPORTANT

Ensure that the MON, MGR and OSD pods are up and running.

19. Scale up the **rook-ceph-operator** and **ocs-operator** deployments.

```
# oc -n openshift-storage scale deployment ocs-operator --replicas=1
```

Verification steps

1. Check the Ceph status to confirm that CephFS is running.

```
# ceph -s
```

Example output:

```
cluster:
  id:     f111402f-84d1-4e06-9fdb-c27607676e55
  health: HEALTH_ERR
          1 filesystem is offline
          1 filesystem is online with fewer MDS than max_mds
          3 daemons have recently crashed
```

```
  services:
    mon: 3 daemons, quorum b,c,a (age 15m)
    mgr: a(active, since 14m)
    mds: ocs-storagecluster-cephfilesystem:0
    osd: 3 osds: 3 up (since 15m), 3 in (since 2h)

  data:
    pools:   3 pools, 96 pgs
    objects: 500 objects, 1.1 GiB
    usage:   5.5 GiB used, 295 GiB / 300 GiB avail
    pgs:     96 active+clean
```

2. Check the Multicloud Object Gateway (MCG) status. It should be active, and the backingstore and bucketclass should be in **Ready** state.

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

IMPORTANT

If the MCG is not in the active state, and the backingstore and bucketclass not in the **Ready** state, you need to restart all the MCG related pods. For more information, see Section 11.1, “Restoring the Multicloud Object Gateway” .

11.1. RESTORING THE MULTICLOUD OBJECT GATEWAY

If the Multicloud Object Gateway (MCG) is not in the active state, and the backingstore and bucketclass is not in the **Ready** state, you need to restart all the MCG related pods, and check the MCG status to confirm that the MCG is back up and running.

Procedure

1. Restart all the pods related to the MCG.

```
# oc delete pods <noobaa-operator> -n openshift-storage
```

```
# oc delete pods <noobaa-core> -n openshift-storage
```

```
# oc delete pods <noobaa-endpoint> -n openshift-storage
```

```
# oc delete pods <noobaa-db> -n openshift-storage
```

<noobaa-operator>

Is the name of the MCG operator

<noobaa-core>

Is the name of the MCG core pod

<noobaa-endpoint>

Is the name of the MCG endpoint

<noobaa-db>

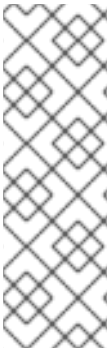
Is the name of the MCG db pod

2. If the RADOS Object Gateway (RGW) is configured, restart the pod.

```
# oc delete pods <rgw-pod> -n openshift-storage
```

<rgw-pod>

Is the name of the RGW pod



NOTE

In OpenShift Container Platform 4.11, after the recovery, RBD PVC fails to get mounted on the application pods. Hence, you need to restart the node that is hosting the application pods. To get the node name that is hosting the application pod, run the following command:

```
# oc get pods <application-pod> -n <namespace> -o yaml | grep nodeName
  nodeName: node_name
```

CHAPTER 12. RESTORING CEPH-MONITOR QUORUM IN OPENSHIFT DATA FOUNDATION

In some circumstances, the **ceph-mons** might lose quorum. If the **mons** cannot form quorum again, there is a manual procedure to get the quorum going again. The only requirement is that at least one **mon** must be healthy. The following steps removes the unhealthy **mons** from quorum and enables you to form a quorum again with a single **mon**, then bring the quorum back to the original size.

For example, if you have three **mons** and lose quorum, you need to remove the two bad **mons** from quorum, notify the good **mon** that it is the only **mon** in quorum, and then restart the good **mon**.

Procedure

1. Stop the **rook-ceph-operator** so that the **mons** are not failed over when you are modifying the **monmap**.

```
# oc -n openshift-storage scale deployment rook-ceph-operator --replicas=0
```

2. Inject a new **monmap**.

WARNING

You must inject the **monmap** very carefully. If run incorrectly, your cluster could be permanently destroyed. The Ceph **monmap** keeps track of the **mon** quorum. The **monmap** is updated to only contain the healthy mon. In this example, the healthy mon is **rook-ceph-mon-b**, while the unhealthy **mons** are **rook-ceph-mon-a** and **rook-ceph-mon-c**.

a. Take a backup of the current **rook-ceph-mon-b** Deployment:

```
# oc -n openshift-storage get deployment rook-ceph-mon-b -o yaml > rook-ceph-mon-b-deployment.yaml
```

b. Open the YAML file and copy the **command** and **arguments** from the **mon** container (see containers list in the following example). This is needed for the **monmap** changes.

```
[...]
  containers:
  - args:
    - --fsid=41a537f2-f282-428e-989f-a9e07be32e47
    - --keyring=/etc/ceph/keyring-store/keyring
    - --log-to-stderr=true
    - --err-to-stderr=true
    - --mon-cluster-log-to-stderr=true
    - '--log-stderr-prefix=debug '
    - --default-log-to-file=false
    - --default-mon-cluster-log-to-file=false
    - --mon-host=$(ROOK_CEPH_MON_HOST)
    - --mon-initial-members=$(ROOK_CEPH_MON_INITIAL_MEMBERS)
```

```
    - --id=b
    - --setuser=ceph
    - --setgroup=ceph
    - --foreground
    - --public-addr=10.100.13.242
    - --setuser-match-path=/var/lib/ceph/mon/ceph-b/store.db
    - --public-bind-addr=$(ROOK_POD_IP)
    command:
    - ceph-mon
[...]
```

c. Cleanup the copied **command** and **args** fields to form a pastable command as follows:

```
# ceph-mon \
    --fsid=41a537f2-f282-428e-989f-a9e07be32e47 \
    --keyring=/etc/ceph/keyring-store/keyring \
    --log-to-stderr=true \
    --err-to-stderr=true \
    --mon-cluster-log-to-stderr=true \
    --log-stderr-prefix=debug \
    --default-log-to-file=false \
    --default-mon-cluster-log-to-file=false \
    --mon-host=$ROOK_CEPH_MON_HOST \
    --mon-initial-members=$ROOK_CEPH_MON_INITIAL_MEMBERS \
    --id=b \
    --setuser=ceph \
    --setgroup=ceph \
    --foreground \
    --public-addr=10.100.13.242 \
    --setuser-match-path=/var/lib/ceph/mon/ceph-b/store.db \
    --public-bind-addr=$ROOK_POD_IP
```

NOTE

Make sure to remove the single quotes around the **--log-stderr-prefix** flag and the parenthesis around the variables being passed **ROOK_CEPH_MON_HOST**, **ROOK_CEPH_MON_INITIAL_MEMBERS** and **ROOK_POD_IP**).

d. Patch the **rook-ceph-mon-b** Deployment to stop the working of this **mon** without deleting the **mon** pod.

```
# oc -n openshift-storage patch deployment rook-ceph-mon-b  --type='json' -p
'[{"op":"remove", "path":"/spec/template/spec/containers/0/livenessProbe"}]'

# oc -n openshift-storage patch deployment rook-ceph-mon-b -p '{"spec": {"template":
{"spec": {"containers": [{"name": "mon", "command": ["sleep", "infinity"], "args": []}]}}}}'
```

e. Perform the following steps on the **mon-b** pod:

i. Connect to the pod of a healthy **mon** and run the following commands:

```
# oc -n openshift-storage exec -it <mon-pod> bash
```

ii. Set the variable.

```
# monmap_path=/tmp/monmap
```

iii. Extract the **monmap** to a file, by pasting the ceph **mon** command from the good **mon** deployment and adding the **--extract-monmap=${monmap_path}** flag.

```
# ceph-mon \
    --fsid=41a537f2-f282-428e-989f-a9e07be32e47 \
    --keyring=/etc/ceph/keyring-store/keyring \
    --log-to-stderr=true \
    --err-to-stderr=true \
    --mon-cluster-log-to-stderr=true \
    --log-stderr-prefix=debug \
    --default-log-to-file=false \
    --default-mon-cluster-log-to-file=false \
    --mon-host=$ROOK_CEPH_MON_HOST \
    --mon-initial-members=$ROOK_CEPH_MON_INITIAL_MEMBERS \
    --id=b \
    --setuser=ceph \
    --setgroup=ceph \
    --foreground \
    --public-addr=10.100.13.242 \
    --setuser-match-path=/var/lib/ceph/mon/ceph-b/store.db \
    --public-bind-addr=$ROOK_POD_IP \
    --extract-monmap=${monmap_path}
```

iv. Review the contents of the **monmap**.

```
# monmaptool --print /tmp/monmap
```

v. Remove the bad **mons** from the **monmap**.

```
# monmaptool ${monmap_path} --rm <bad_mon>
```

In this example we remove **mon0** and **mon2**:

```
# monmaptool ${monmap_path} --rm a
# monmaptool ${monmap_path} --rm c
```

vi. Inject the modified **monmap** into the good **mon**, by pasting the ceph **mon** command and adding the **--inject-monmap=${monmap_path}** flag as follows:

```
# ceph-mon \
    --fsid=41a537f2-f282-428e-989f-a9e07be32e47 \
    --keyring=/etc/ceph/keyring-store/keyring \
    --log-to-stderr=true \
    --err-to-stderr=true \
    --mon-cluster-log-to-stderr=true \
    --log-stderr-prefix=debug \
    --default-log-to-file=false \
    --default-mon-cluster-log-to-file=false \
    --mon-host=$ROOK_CEPH_MON_HOST \
    --mon-initial-members=$ROOK_CEPH_MON_INITIAL_MEMBERS \
```

```
--id=b \
--setuser=ceph \
--setgroup=ceph \
--foreground \
--public-addr=10.100.13.242 \
--setuser-match-path=/var/lib/ceph/mon/ceph-b/store.db \
--public-bind-addr=$ROOK_POD_IP \
--inject-monmap=${monmap_path}
```

vii. Exit the shell to continue.

3. Edit the Rook **configmaps**.

   a. Edit the **configmap** that the operator uses to track the **mons**.

   ```
   # oc -n openshift-storage edit configmap rook-ceph-mon-endpoints
   ```

   b. Verify that in the data element you see three **mons** such as the following (or more depending on your **moncount**):

   ```
   data: a=10.100.35.200:6789;b=10.100.13.242:6789;c=10.100.35.12:6789
   ```

   c. Delete the bad **mons** from the list to end up with a single good **mon**. For example:

   ```
   data: b=10.100.13.242:6789
   ```

   d. Save the file and exit.

   e. Now, you need to adapt a **Secret** which is used for the **mons** and other components.

      i. Set a value for the variable **good_mon_id**.
      For example:

      ```
      # good_mon_id=b
      ```

      ii. You can use the **oc patch** command to patch the **rook-ceph-config** secret and update the two key/value pairs **mon_host** and **mon_initial_members**.

      ```
      # mon_host=$(oc -n openshift-storage get svc rook-ceph-mon-b -o
      jsonpath='{.spec.clusterIP}')

      # oc -n openshift-storage patch secret rook-ceph-config -p '{"stringData":
      {"mon_host": "[v2:"${mon_host}":3300,v1:"${mon_host}":6789]",
      "mon_initial_members": ""${good_mon_id}""}}'
      ```

NOTE

If you are using **hostNetwork: true**, you need to replace the **mon_host** var with the node IP the **mon** is pinned to (**nodeSelector**). This is because there is no **rook-ceph-mon-*** service created in that "mode".

4. Restart the **mon**.

You need to restart the good **mon** pod with the original **ceph-mon** command to pick up the changes.

a. Use the **oc replace** command on the backup of the **mon** deployment YAML file:

```
# oc replace --force -f rook-ceph-mon-b-deployment.yaml
```

NOTE

Option **--force** deletes the deployment and creates a new one.

b. Verify the status of the cluster.
The status should show one **mon** in quorum. If the status looks good, your cluster should be healthy again.

5. Delete the two mon deployments that are no longer expected to be in quorum.
For example:

```
# oc delete deploy <rook-ceph-mon-1>
# oc delete deploy <rook-ceph-mon-2>
```

In this example the deployments to be deleted are **rook-ceph-mon-a** and **rook-ceph-mon-c**.

6. Restart the operator.

a. Start the rook operator again to resume monitoring the health of the cluster.

NOTE

It is safe to ignore the errors that a number of resources already exist.

```
# oc -n openshift-storage scale deployment rook-ceph-operator --replicas=1
```

The operator automatically adds more **mons** to increase the quorum size again depending on the **mon** count.

CHAPTER 13. ENABLING THE RED HAT OPENSHIFT DATA FOUNDATION CONSOLE PLUGIN

The Data Foundation console plugin is enabled by default. In case, this option was unchecked during OpenShift Data Foundation Operator installation, use the following instructions to enable the console plugin post-deployment either from the graphical user interface (GUI) or command-line interface.

Prerequisites

- You have administrative access to the OpenShift Web Console.
- OpenShift Data Foundation Operator is installed and running in the **openshift-storage** namespace.

Procedure

From user interface

1. In the OpenShift Web Console, click **Operators → Installed Operators** to view all the installed operators.
2. Ensure that the **Project** selected is **openshift-storage**.
3. Click on the **OpenShift Data Foundation** operator.
4. Enable the console plugin option.
    a. In the **Details** tab, click the **pencil** icon under the **Console plugin**.
    b. Select **Enable**, and click **Save**.

From command-line interface

- Execute the following command to enable the console plugin option:

```
$ oc patch console.operator cluster -n openshift-storage --type json -p '[{"op": "add", "path": "/spec/plugins", "value": ["odf-console"]}]'
```

Verification steps

- After the console plugin option is enabled, a pop-up with a message, **Web console update is available** appears on the GUI. Click **Refresh web console** from this pop-up for the console changes to reflect.
    - In the Web Console, navigate to **Storage** and verify if **Data Foundation** is available.

CHAPTER 14. CHANGING RESOURCES FOR THE OPENSHIFT DATA FOUNDATION COMPONENTS

When you install OpenShift Data Foundation, it comes with pre-defined resources that the OpenShift Data Foundation pods can consume. In some situations with higher I/O load, it might be required to increase these limits.

- To change the CPU and memory resources on the rook-ceph pods, see Section 14.1, "Changing the CPU and memory resources on the rook-ceph pods".
- To tune the resources for the Multicloud Object Gateway (MCG), see Section 14.2, "Tuning the resources for the MCG".

14.1. CHANGING THE CPU AND MEMORY RESOURCES ON THE ROOK-CEPH PODS

When you install OpenShift Data Foundation, it comes with pre-defined CPU and memory resources for the rook-ceph pods. You can manually increase these values according to the requirements.

You can change the CPU and memory resources on the following pods:

- **mgr**
- **mds**
- **rgw**

The following example illustrates how to change the CPU and memory resources on the rook-ceph pods. In this example, the existing MDS pod values of **cpu** and **memory** are increased from **1** and **4Gi** to **2** and **8Gi** respectively.

1. Edit the storage cluster:

```
# oc edit storagecluster -n openshift-storage <storagecluster_name>
```

<storagecluster_name>

Specify the name of the storage cluster.
For example:

```
# oc edit storagecluster -n openshift-storage ocs-storagecluster
```

2. Add the following lines to the storage cluster Custom Resource (CR):

```
spec:
  resources:
    mds:
      limits:
        cpu: 2
        memory: 8Gi
      requests:
        cpu: 2
        memory: 8Gi
```

3. Save the changes and exit the editor.

4. Alternatively, run the **oc patch** command to change the CPU and memory value of the **mds** pod:

```
# oc patch -n openshift-storage storagecluster <storagecluster_name>
    --type merge \
    --patch '{"spec": {"resources": {"mds": {"limits": {"cpu": "2","memory": "8Gi"},"requests":
{"cpu": "2","memory": "8Gi"}}}}}'
```

<storagecluster_name>

Specify the name of the storage cluster.
For example:

```
# oc patch -n openshift-storage storagecluster ocs-storagecluster \
    --type merge \
    --patch '{"spec": {"resources": {"mds": {"limits": {"cpu": "2","memory": "8Gi"},"requests":
{"cpu": "2","memory": "8Gi"}}}}}'
```

14.2. TUNING THE RESOURCES FOR THE MCG

The default configuration for the Multicloud Object Gateway (MCG) is optimized for low resource consumption and not performance. For more information on how to tune the resources for the MCG, see the Red Hat Knowledgebase solution Performance tuning guide for Multicloud Object Gateway (NooBaa).

CHAPTER 15. DISABLING MULTICLOUD OBJECT GATEWAY EXTERNAL SERVICE AFTER DEPLOYING OPENSHIFT DATA FOUNDATION

When you deploy OpenShift Data Foundation, public IPs are created even when OpenShift is installed as a private cluster. However, you can disable the Multicloud Object Gateway (MCG) load balancer usage by using the **disableLoadBalancerService** variable in the storagecluster CRD. This restricts MCG from creating any public resources for private clusters and helps to disable the NooBaa service **EXTERNAL-IP**.

Procedure

- Run the following command and add the **disableLoadBalancerService** variable in the storagecluster YAML to set the service to ClusterIP:

```
$ oc edit storagecluster -n openshift-storage <storagecluster_name>
[...]
spec:
  arbiter: {}
  encryption:
    kms: {}
  externalStorage: {}
  managedResources:
    cephBlockPools: {}
    cephCluster: {}
    cephConfig: {}
    cephDashboard: {}
    cephFilesystems: {}
    cephNonResilientPools: {}
    cephObjectStoreUsers: {}
    cephObjectStores: {}
    cephRBDMirror: {}
    cephToolbox: {}
  mirroring: {}
  multiCloudGateway:
    disableLoadBalancerService: true        <---------- Add this
    endpoints:
[...]
```

NOTE

To undo the changes and set the service to LoadBalancer, set the **disableLoadBalancerService** variable to **false** or remove that line completely.

CHAPTER 16. ACCESSING ODF-CONSOLE WITH THE OVS-MULTITENANT PLUGIN BY MANUALLY ENABLING GLOBAL POD NETWORKING

In OpenShift Container Platform, when **ovs-multitenant** plugin is used for software-defined networking (SDN), pods from different projects cannot send packets to or receive packets from pods and services of a different project. By default, pods can not communicate between namespaces or projects because a project's pod networking is not global.

To access odf-console, the OpenShift console pod in the **openshift-console** namespace needs to connect with the OpenShift Data Foundation odf-console in the **openshift-storage** namespace. This is possible only when you manually enable global pod networking.

Issue

- When`ovs-multitenant` plugin is used in the OpenShift Container Platform, the odf-console plugin fails with the following message:

```
GET request for "odf-console" plugin failed: Get "https://odf-console-service.openshift-storage.svc.cluster.local:9001/locales/en/plugin__odf-console.json": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
```

Resolution

- Make the pod networking for the OpenShift Data Foundation project global:

```
$ oc adm pod-network make-projects-global openshift-storage
```

CHAPTER 17. ANNOTATING ENCRYPTED RBD STORAGE CLASSES

Starting with OpenShift Data Foundation 4.14, when the OpenShift console creates a RADOS block device (RBD) storage class with encryption enabled, the annotation is set automatically. However, you need to add the annotation, **cdi.kubevirt.io/clone-strategy=copy** for any of the encrypted RBD storage classes that were previously created before updating to the OpenShift Data Foundation version 4.14. This enables customer data integration (CDI) to use host-assisted cloning instead of the default smart cloning.

The keys used to access an encrypted volume are tied to the namespace where the volume was created. When cloning an encrypted volume to a new namespace, such as, provisioning a new OpenShift Virtualization virtual machine, a new volume must be created and the content of the source volume must then be copied into the new volume. This behavior is triggered automatically if the storage class is properly annotated.