Red Hat Ceph Storage 5

Configuration Guide

Configuration settings for Red Hat Ceph Storage
Configuration settings for Red Hat Ceph Storage
Abstract

This document provides instructions for configuring Red Hat Ceph Storage at boot time and run time. It also provides configuration reference information. 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.
# Table of Contents

## CHAPTER 1. THE BASICS OF CEPH CONFIGURATION

- 1.1. PREREQUISITES ......................................................... 4
- 1.2. CEPH CONFIGURATION .................................................. 4
- 1.3. THE CEPH CONFIGURATION DATABASE .......................... 4
- 1.4. USING THE CEPH METAVARIABLES ............................... 6
- 1.5. VIEWING THE CEPH CONFIGURATION AT RUNTIME ............ 7
- 1.6. VIEWING A SPECIFIC CONFIGURATION AT RUNTIME ........... 7
- 1.7. SETTING A SPECIFIC CONFIGURATION AT RUNTIME .......... 8
- 1.8. OSD MEMORY TARGET .................................................. 9
- 1.8.1. Setting the OSD memory target ................................ 10
- 1.9. AUTOMATICALLY TUNING OSD MEMORY ......................... 11
- 1.10. MDS MEMORY CACHE LIMIT ........................................ 13
- 1.11. ADDITIONAL RESOURCES ............................................ 13

## CHAPTER 2. CEPH NETWORK CONFIGURATION

- 2.1. PREREQUISITES ......................................................... 14
- 2.2. NETWORK CONFIGURATION FOR CEPH ............................ 14
- 2.3. CEPH NETWORK MESSENGER ........................................ 16
- 2.4. CONFIGURING A PUBLIC NETWORK ............................... 17
- 2.5. CONFIGURING A PRIVATE NETWORK ............................... 18
- 2.6. VERIFY THE FIREWALL SETTINGS ................................ 19
- 2.7. FIREWALL SETTINGS FOR CEPH MONITOR NODE ............. 19
- 2.8. FIREWALL SETTINGS FOR CEPH OSDS ............................ 20
- 2.9. ADDITIONAL RESOURCES ............................................ 21

## CHAPTER 3. CEPH MONITOR CONFIGURATION

- 3.1. PREREQUISITES ......................................................... 23
- 3.2. CEPH MONITOR CONFIGURATION .................................... 23
- 3.2.1. Accessing the configuration database ........................... 23
- 3.3. CEPH CLUSTER MAPS .................................................. 24
- 3.4. CEPH MONITOR QUORUM ............................................. 25
- 3.5. CEPH MONITOR CONSISTENCY ...................................... 25
- 3.6. BOOTSTRAP THE CEPH MONITOR .................................. 26
- 3.7. CEPH MONITOR SECTION IN THE CONFIGURATION FILE ....... 26
- 3.8. MINIMUM CONFIGURATION FOR A CEPH MONITOR .......... 26
- 3.9. UNIQUE IDENTIFIER FOR CEPH .................................... 27
- 3.10. CEPH MONITOR DATA STORE ...................................... 28
- 3.11. CEPH STORAGE CAPACITY .......................................... 28
- 3.12. CEPH HEARTBEAT .................................................... 29
- 3.13. CEPH MONITOR SYNCHRONIZATION ROLE .................... 29
- 3.14. CEPH TIME SYNCHRONIZATION .................................. 30
- 3.15. ADDITIONAL RESOURCES .......................................... 31

## CHAPTER 4. CEPH AUTHENTICATION CONFIGURATION

- 4.1. PREREQUISITES ......................................................... 32
- 4.2. CEPHX AUTHENTICATION ............................................. 32
- 4.3. ENABLING CEPHX .................................................... 32
- 4.4. DISABLING CEPHX .................................................... 34
- 4.5. CEPHX USER KEYRINGS ............................................. 35
- 4.6. CEPHX DAEMON KEYRINGS ........................................ 35
- 4.7. CEPHX MESSAGE SIGNATURES ..................................... 35
- 4.8. ADDITIONAL RESOURCES ............................................ 36
<table>
<thead>
<tr>
<th>Chapter Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. POOLS, PLACEMENT GROUPS, AND CRUSH CONFIGURATION</td>
<td>37</td>
</tr>
<tr>
<td>5.1. PREREQUISITES</td>
<td>37</td>
</tr>
<tr>
<td>5.2. POOLS PLACEMENT GROUPS AND CRUSH</td>
<td>37</td>
</tr>
<tr>
<td>5.3. ADDITIONAL RESOURCES</td>
<td>37</td>
</tr>
<tr>
<td>6. CEPH OBJECT STORAGE DAEMON (OSD) CONFIGURATION</td>
<td>38</td>
</tr>
<tr>
<td>6.1. PREREQUISITES</td>
<td>38</td>
</tr>
<tr>
<td>6.2. CEPH OSD CONFIGURATION</td>
<td>38</td>
</tr>
<tr>
<td>6.3. SCRUBBING THE OSD</td>
<td>38</td>
</tr>
<tr>
<td>6.4. BACKFILLING AN OSD</td>
<td>39</td>
</tr>
<tr>
<td>6.5. OSD RECOVERY</td>
<td>39</td>
</tr>
<tr>
<td>6.6. ADDITIONAL RESOURCES</td>
<td>39</td>
</tr>
<tr>
<td>7. CEPH MONITOR AND OSD INTERACTION CONFIGURATION</td>
<td>40</td>
</tr>
<tr>
<td>7.1. PREREQUISITES</td>
<td>40</td>
</tr>
<tr>
<td>7.2. CEPH MONITOR AND OSD INTERACTION</td>
<td>40</td>
</tr>
<tr>
<td>7.3. OSD HEARTBEAT</td>
<td>40</td>
</tr>
<tr>
<td>7.4. REPORTING AN OSD AS DOWN</td>
<td>41</td>
</tr>
<tr>
<td>7.5. REPORTING A PEERING FAILURE</td>
<td>42</td>
</tr>
<tr>
<td>7.6. OSD REPORTING STATUS</td>
<td>43</td>
</tr>
<tr>
<td>7.7. ADDITIONAL RESOURCES</td>
<td>43</td>
</tr>
<tr>
<td>8. CEPH DEBUGGING AND LOGGING CONFIGURATION</td>
<td>45</td>
</tr>
<tr>
<td>8.1. PREREQUISITES</td>
<td>45</td>
</tr>
<tr>
<td>8.2. ADDITIONAL RESOURCES</td>
<td>45</td>
</tr>
<tr>
<td>Appendix A. GENERAL CONFIGURATION OPTIONS</td>
<td>46</td>
</tr>
<tr>
<td>Appendix B. CEPH NETWORK CONFIGURATION OPTIONS</td>
<td>48</td>
</tr>
<tr>
<td>Appendix C. CEPH MONITOR CONFIGURATION OPTIONS</td>
<td>54</td>
</tr>
<tr>
<td>Appendix D. CEPHX CONFIGURATION OPTIONS</td>
<td>70</td>
</tr>
<tr>
<td>Appendix E. POOLS, PLACEMENT GROUPS, AND CRUSH CONFIGURATION OPTIONS</td>
<td>74</td>
</tr>
<tr>
<td>Appendix F. OBJECT STORAGE DAEMON (OSD) CONFIGURATION OPTIONS</td>
<td>80</td>
</tr>
<tr>
<td>Appendix G. CEPH MONITOR AND OSD CONFIGURATION OPTIONS</td>
<td>95</td>
</tr>
<tr>
<td>Appendix H. CEPH SCRUBBING OPTIONS</td>
<td>100</td>
</tr>
<tr>
<td>Appendix I. BLUESTORE CONFIGURATION OPTIONS</td>
<td>104</td>
</tr>
</tbody>
</table>
CHAPTER 1. THE BASICS OF CEPH CONFIGURATION

As a storage administrator, you need to have a basic understanding of how to view the Ceph configuration, and how to set the Ceph configuration options for the Red Hat Ceph Storage cluster. You can view and set the Ceph configuration options at runtime.

1.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

1.2. CEPH CONFIGURATION

All Red Hat Ceph Storage clusters have a configuration, which defines:

- Cluster Identity
- Authentication settings
- Ceph daemons
- Network configuration
- Node names and addresses
- Paths to keyrings
- Paths to OSD log files
- Other runtime options

A deployment tool, such as cephadm, will typically create an initial Ceph configuration file for you. However, you can create one yourself if you prefer to bootstrap a Red Hat Ceph Storage cluster without using a deployment tool.

Additional Resources

- For more information about cephadm and the Ceph orchestrator, see the Red Hat Ceph Storage Operations Guide.

1.3. THE CEPH CONFIGURATION DATABASE

The Ceph Monitor manages a configuration database of Ceph options that centralize configuration management by storing configuration options for the entire storage cluster. By centralizing the Ceph configuration in a database, this simplifies storage cluster administration.

The overall priority order that Ceph uses to set options is:

- Compiled-in default values
- Ceph cluster configuration database
- Local ceph.conf file
- Runtime override, using the ceph daemon DAEMON-NAME config set or ceph tell DAEMON-NAME injectargs commands
There are still a few Ceph options that can be defined in the local Ceph configuration file, which is `/etc/ceph/ceph.conf` by default. However, `ceph.conf` has been deprecated for Red Hat Ceph Storage 5.

`cephadm` uses a basic `ceph.conf` file that only contains a minimal set of options for connecting to Ceph Monitors, authenticating, and fetching configuration information. In most cases, `cephadm` uses only the `mon_host` option. To avoid using `ceph.conf` only for the `mon_host` option, use DNS SRV records to perform operations with Monitors.

**IMPORTANT**

Red Hat recommends that you use the `assimilate-conf` administrative command to move valid options into the configuration database from the `ceph.conf` file. For more information about `assimilate-conf`, see Administrative Commands.

Ceph allows you to make changes to the configuration of a daemon at runtime. This capability can be useful for increasing or decreasing the logging output, by enabling or disabling debug settings, and can even be used for runtime optimization.

**NOTE**

When the same option exists in the configuration database and the Ceph configuration file, the configuration database option has a lower priority than what is set in the Ceph configuration file.

**Sections and Masks**

Just as you can configure Ceph options globally, per daemon type, or by a specific daemon in the Ceph configuration file, you can also configure the Ceph options in the configuration database according to these sections. Ceph configuration options can have a mask associated with them. These masks can further restrict which daemons or clients the options apply to.

Masks have two forms:

**type:location**

The `type` is a CRUSH property, for example, `rack` or `host`. The `location` is a value for the property type. For example, `host:foo` limits the option only to daemons or clients running on a particular node, `foo` in this example.

**class:device-class**

The `device-class` is the name of the CRUSH device class, such as `hdd` or `ssd`. For example, `class:ssd` limits the option only to Ceph OSDs backed by solid state drives (SSD). This mask has no effect on non-OSD daemons of clients.

**Administrative Commands**

The Ceph configuration database can be administered with the subcommand `ceph config ACTION`.

These are the actions you can do:

**dump**

Dumps the entire configuration database of options for the storage cluster.

**get WHO**

Dumps the configuration for a specific daemon or client. For example, `WHO` can be a daemon, like `mds.a`.

**set WHO OPTION VALUE**
Sets a configuration option in the Ceph configuration database.

show WHO

Shows the reported running configuration for a running daemon. These options might be different from those stored by the Ceph Monitors if there is a local configuration file in use or options have been overridden on the command line or at run time. Also, the source of the option values is reported as part of the output.

assimilate-conf -i INPUT_FILE -o OUTPUT_FILE

Assimilate a configuration file from the INPUT_FILE and move any valid options into the Ceph Monitors’ configuration database. Any options that are unrecognized, invalid, or cannot be controlled by the Ceph Monitor return in an abbreviated configuration file stored in the OUTPUT_FILE. This command can be useful for transitioning from legacy configuration files to a centralized configuration database. Note that when you assimilate a configuration and the Monitors or other daemons have different configuration values set for the same set of options, the end result depends on the order in which the files are assimilated.

help OPTION -f json-pretty

Displays help for a particular OPTION using a JSON-formatted output.

Additional Resources

- For more information about the command, see Setting a specific configuration at runtime.

1.4. USING THE CEPH METAVARIABLES

Metavariables simplify Ceph storage cluster configuration dramatically. When a metavariable is set in a configuration value, Ceph expands the metavariable into a concrete value.

Metavariables are very powerful when used within the [global], [osd], [mon], or [client] sections of the Ceph configuration file. However, you can also use them with the administration socket. Ceph metavariables are similar to Bash shell expansion.

Ceph supports the following metavariables:

$cluster

Description

Expands to the Ceph storage cluster name. Useful when running multiple Ceph storage clusters on the same hardware.

Example

/etc/ceph/$cluster.keyring

Default

ceph

$type

Description

Expands to one of osd or mon, depending on the type of the instant daemon.

Example

/var/lib/ceph/$type

$id

Description
Expands to the daemon identifier. For osd.0, this would be 0.

Example
/var/lib/ceph/$type/$cluster-$id

$host
Description
Expands to the host name of the instant daemon.

$name
Description
Expands to $type.$id.

Example
/var/run/ceph/$cluster-$name.asok

1.5. VIEWING THE CEPH CONFIGURATION AT RUNTIME

The Ceph configuration files can be viewed at boot time and run time.

Prerequisites

- Root-level access to the Ceph node.
- Access to admin keyring.

Procedure

1. To view a runtime configuration, log in to a Ceph node running the daemon and execute:

   Syntax
   
   ```
   ceph daemon DAEMON_TYPE.ID config show
   ```

   To see the configuration for osd.0, you can log into the node containing osd.0 and execute this command:

   Example
   
   ```
   [root@osd ~]# ceph daemon osd.0 config show
   ```

2. For additional options, specify a daemon and help.

   Example
   
   ```
   [root@osd ~]# ceph daemon osd.0 help
   ```

1.6. VIEWING A SPECIFIC CONFIGURATION AT RUNTIME
Configuration settings for Red Hat Ceph Storage can be viewed at runtime from the Ceph Monitor node.

**Prerequisites**

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor node.

**Procedure**

1. Log into a Ceph node and execute:

   **Syntax**
   ```
   ceph daemon DAEMON_TYPE.ID config get PARAMETER
   ```

   **Example**
   ```
   [root@mon ~]# ceph daemon osd.0 config get public_addr
   ```

**1.7. SETTING A SPECIFIC CONFIGURATION AT RUNTIME**

To set a specific Ceph configuration at runtime, use the `ceph config set` command.

**Prerequisites**

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor or OSD nodes.

**Procedure**

1. Set the configuration on all Monitor or OSD daemons:

   **Syntax**
   ```
   ceph config set DAEMON CONFIG-OPTION VALUE
   ```

   **Example**
   ```
   [root@mon ~]# ceph config set osd debug_osd 10
   ```

2. Validate that the option and value are set:

   **Example**
   ```
   [root@mon ~]# ceph config dump
   osd      advanced debug_osd  10/10
   ```

   - To remove the configuration option from all daemons:
Syntax  
```bash
ceph config rm DAEMON CONFIG-OPTION VALUE
```

**Example**
```
[root@mon ~]# ceph config rm osd debug_osd
```

- To set the configuration for a specific daemon:

Syntax  
```bash
ceph config set DAEMON.DAEMON-NUMBER CONFIG-OPTION VALUE
```

**Example**
```
[root@mon ~]# ceph config set osd.0 debug_osd 10
```

- To validate that the configuration is set for the specified daemon:

**Example**
```
[root@mon ~]# ceph config dump
osd.0      advanced debug_osd     10/10
```

- To remove the configuration for a specific daemon:

Syntax  
```bash
ceph config rm DAEMON.DAEMON-NUMBER CONFIG-OPTION
```

**Example**
```
[root@mon ~]# ceph config rm osd.0 debug_osd
```

**NOTE**

If you use a client that does not support reading options from the configuration database, or if you still need to use `ceph.conf` to change your cluster configuration for other reasons, run the following command:

```
ceph config set mgr mgr/cephadm/manage_etc_ceph_ceph_conf false
```

You must maintain and distribute the `ceph.conf` file across the storage cluster.

### 1.8. OSD MEMORY TARGET

BlueStore keeps OSD heap memory usage under a designated target size with the `osd_memory_target` configuration option.
The option `osd_memory_target` sets OSD memory based upon the available RAM in the system. Use this option when TCMalloc is configured as the memory allocator, and when the `bluestore_cache_autotune` option in BlueStore is set to `true`.

Ceph OSD memory caching is more important when the block device is slow; for example, traditional hard drives, because the benefit of a cache hit is much higher than it would be with a solid state drive. However, this must be weighed into a decision to colocate OSDs with other services, such as in a hyper-converged infrastructure (HCI) or other applications.

### 1.8.1. Setting the OSD memory target

Use the `osd_memory_target` option to set the maximum memory threshold for all OSDs in the storage cluster, or for specific OSDs. An OSD with an `osd_memory_target` option set to 16 GB may use up to 16 GB of memory.

![NOTE](image.png)

**NOTE**

Configuration options for individual OSDs take precedence over the settings for all OSDs.

**Prerequisites**

- A running Red Hat Ceph Storage cluster.
- Root-level access to all hosts in the storage cluster.

**Procedure**

- To set `osd_memory_target` for all OSDs in the storage cluster:

  Syntax

  ```
  ceph config set osd osd_memory_target VALUE
  ```

  `VALUE` is the number of GBytes of memory to be allocated to each OSD in the storage cluster.

- To set `osd_memory_target` for a specific OSD in the storage cluster:

  Syntax

  ```
  ceph config set osd.id osd_memory_target VALUE
  ```

  `.id` is the ID of the OSD and `VALUE` is the number of GBytes of memory to be allocated to the specified OSD. For example, to configure the OSD with ID 8 to use up to 16 GBytes of memory:

  **Example**

  ```
  [root@mon ~]# ceph config set osd.8 osd_memory_target 16G
  ```

  - To set an individual OSD to use one maximum amount of memory and configure the rest of the OSDs to use another amount, specify the individual OSD first:

  **Example**

  ```
```
1.9. AUTOMATICALLY TUNING OSD MEMORY

The OSD daemons adjust the memory consumption based on the `osd_memory_target` configuration option. The option `osd_memory_target` sets OSD memory based upon the available RAM in the system.

If Red Hat Ceph Storage is deployed on dedicated nodes that do not share memory with other services, `cephadm` automatically adjusts the per-OSD consumption based on the total amount of RAM and the number of deployed OSDs.

**IMPORTANT**

By default, the `osd_memory_target_autotune` parameter is set to `true` in Red Hat Ceph Storage 5.1.

**Syntax**

```bash
ceph config set osd osd_memory_target_autotune true
```

Once the storage cluster is upgraded to Red Hat Ceph Storage 5.0, for cluster maintenance such as addition of OSDs or replacement of OSDs, Red Hat recommends setting `osd_memory_target_autotune` parameter to `true` to autotune osd memory as per system memory.

Cephadm starts with a fraction `mgr/cephadm/autotune_memory_target_ratio`, which defaults to `0.7` of the total RAM in the system, subtract off any memory consumed by non-autotuned daemons such as non-OSDS and for OSDs for which `osd_memory_target_autotune` is false, and then divide by the remaining OSDs.

The `osd_memory_target` parameter is calculated as follows:

**Syntax**

```bash
osd_memory_target = TOTAL_RAM_OF_THE_OSD * (1048576) * (autotune_memory_target_ratio) / NUMBER_OF_OSDS_IN_THE_OSD_NODE - (SPACE_ALLOCATED_FOR_OTHER_DAEMONS)
```

`SPACE_ALLOCATED_FOR_OTHER_DAEMONS` may optionally include the following daemon space allocations:

- Alertmanager: 1 GB
- Grafana: 1 GB
- Ceph Manager: 4 GB
- Ceph Monitor: 2 GB
- Node-exporter: 1 GB
- Prometheus: 1 GB

For example, if a node has 24 OSDs and has 251 GB RAM space, then `osd_memory_target` is 7860684936.
The final targets are reflected in the configuration database with options. You can view the limits and the current memory consumed by each daemon from the `ceph orch ps` output under **MEM LIMIT** column.

**NOTE**

In Red Hat Ceph Storage 5.1, the default setting of `osd_memory_target_autotune true` is unsuitable for hyperconverged infrastructures where compute and Ceph storage services are colocated. In a hyperconverged infrastructure, the `autotune_memory_target_ratio` can be set to **0.2** to reduce the memory consumption of Ceph.

**Example**

```
[ceph: root@host01 ]# ceph config set mgr
mgr/cephadm/autotune_memory_target_ratio 0.2
```

You can manually set a specific memory target for an OSD in the storage cluster.

**Example**

```
[ceph: root@host01 ]# ceph config set osd.123 osd_memory_target 7860684936
```

You can manually set a specific memory target for an OSD host in the storage cluster.

**Syntax**

```
ceph config set osd/host:HOSTNAME osd_memory_target _TARGET_BYTES_
```

**Example**

```
[ceph: root@host01 ]# ceph config set osd/host:host01 osd_memory_target 1000000000
```

**NOTE**

Enabling `osd_memory_target_autotune` overwrites existing manual OSD memory target settings. To prevent daemon memory from being tuned even when the `osd_memory_target_autotune` option or other similar options are enabled, set the `_no_autotune_memory` label on the host.

**Syntax**

```
ceph orch host label add HOSTNAME _no_autotune_memory
```

You can exclude an OSD from memory autotuning by disabling the autotune option and setting a specific memory target.

**Example**

```
[ceph: root@host01 ]# ceph config set osd.123 osd_memory_target_autotune false
[ceph: root@host01 ]# ceph config set osd.123 osd_memory_target 16G
```
1.10. MDS MEMORY CACHE LIMIT

MDS servers keep their metadata in a separate storage pool, named `cephfs_metadata`, and are the users of Ceph OSDs. For Ceph File Systems, MDS servers have to support an entire Red Hat Ceph Storage cluster, not just a single storage device within the storage cluster, so their memory requirements can be significant, particularly if the workload consists of small-to-medium-size files, where the ratio of metadata to data is much higher.

Example: Set the `mds_cache_memory_limit` to 2000000000 bytes

```plaintext
ceph_conf_overrides:
  osd:
    mds_cache_memory_limit=2000000000
```

NOTE

For a large Red Hat Ceph Storage cluster with a metadata-intensive workload, do not put an MDS server on the same node as other memory-intensive services, doing so gives you the option to allocate more memory to MDS, for example, sizes greater than 100 GB.

Additional Resources

- See Metadata Server cache size limits in Red Hat Ceph Storage File System Guide.

1.11. ADDITIONAL RESOURCES

- See the general Ceph configuration options in Appendix A for specific option descriptions and usage.
CHAPTER 2. CEPH NETWORK CONFIGURATION

As a storage administrator, you must understand the network environment that the Red Hat Ceph Storage cluster will operate in, and configure the Red Hat Ceph Storage accordingly. Understanding and configuring the Ceph network options will ensure optimal performance and reliability of the overall storage cluster.

2.1. PREREQUISITES

- Network connectivity.
- Installation of the Red Hat Ceph Storage software.

2.2. NETWORK CONFIGURATION FOR CEPH

Network configuration is critical for building a high performance Red Hat Ceph Storage cluster. The Ceph storage cluster does not perform request routing or dispatching on behalf of the Ceph client. Instead, Ceph clients make requests directly to Ceph OSD daemons. Ceph OSDs perform data replication on behalf of Ceph clients, which means replication and other factors impose additional loads on the networks of Ceph storage clusters.

Ceph has one network configuration requirement that applies to all daemons. The Ceph configuration file must specify the host for each daemon.

Some deployment utilities, such as cephadm creates a configuration file for you. Do not set these values if the deployment utility does it for you.

**IMPORTANT**

The host option is the short name of the node, not its FQDN. It is not an IP address.

All Ceph clusters must use a public network. However, unless you specify an internal cluster network, Ceph assumes a single public network. Ceph can function with a public network only, but for large storage clusters, you will see significant performance improvement with a second private network for carrying only cluster-related traffic.

**IMPORTANT**

Red Hat recommends running a Ceph storage cluster with two networks. One public network and one private network.

To support two networks, each Ceph Node will need to have more than one network interface card (NIC).
There are several reasons to consider operating two separate networks:

- **Performance:** Ceph OSDs handle data replication for the Ceph clients. When Ceph OSDs replicate data more than once, the network load between Ceph OSDs easily dwarfs the network load between Ceph clients and the Ceph storage cluster. This can introduce latency and create a performance problem. Recovery and rebalancing can also introduce significant latency on the public network.

- **Security:** While most people are generally civil, some actors will engage in what is known as a Denial of Service (DoS) attack. When traffic between Ceph OSDs gets disrupted, peering may fail and placement groups may no longer reflect an **active + clean** state, which may prevent users from reading and writing data. A great way to defeat this type of attack is to maintain a completely separate cluster network that does not connect directly to the internet.

Network configuration settings are not required. Ceph can function with a public network only, assuming a public network is configured on all hosts running a Ceph daemon. However, Ceph allows you to establish much more specific criteria, including multiple IP networks and subnet masks for your public network. You can also establish a separate cluster network to handle OSD heartbeat, object replication, and recovery traffic.

Do not confuse the IP addresses you set in the configuration with the public-facing IP addresses network clients might use to access your service. Typical internal IP networks are often **192.168.0.0** or **10.0.0.0**.

**NOTE**

Ceph uses CIDR notation for subnets, for example, **10.0.0.0/24**.
IMPORTANT

If you specify more than one IP address and subnet mask for either the public or the private network, the subnets within the network must be capable of routing to each other. Additionally, make sure you include each IP address and subnet in your IP tables and open ports for them as necessary.

When you configured the networks, you can restart the cluster or restart each daemon. Ceph daemons bind dynamically, so you do not have to restart the entire cluster at once if you change the network configuration.

Additional Resources

- See the common options in *Red Hat Ceph Storage Configuration Guide, Appendix B* for specific option descriptions and usage.

### 2.3. CEPH NETWORK MESSENGER

Messenger is the Ceph network layer implementation. Red Hat supports two messenger types:

- **simple**
- **async**

In Red Hat Ceph Storage 4 and higher, **async** is the default messenger type. To change the messenger type, specify the `ms_type` configuration setting in the `[global]` section of the Ceph configuration file.

**NOTE**

For the **async** messenger, Red Hat supports the **posix** transport type, but does not currently support **rdma** or **dpdk**. By default, the `ms_type` setting in Red Hat Ceph Storage 4 or higher reflects **async+posix**, where **async** is the messenger type and **posix** is the transport type.

**SimpleMessenger**

The **SimpleMessenger** implementation uses TCP sockets with two threads per socket. Ceph associates each logical session with a connection. A pipe handles the connection, including the input and output of each message. While **SimpleMessenger** is effective for the **posix** transport type, it is not effective for other transport types such as **rdma** or **dpdk**.

**AsyncMessenger**

Consequently, **AsyncMessenger** is the default messenger type for Red Hat Ceph Storage 4 or higher. For Red Hat Ceph Storage 4 or higher, the **AsyncMessenger** implementation uses TCP sockets with a fixed-size thread pool for connections, which should be equal to the highest number of replicas or erasure-code chunks. The thread count can be set to a lower value if performance degrades due to a low CPU count or a high number of OSDs per server.

**NOTE**

Red Hat does not support other transport types such as **rdma** or **dpdk** at this time.
• See the AsyncMessenger options in Red Hat Ceph Storage Configuration Guide, Appendix B for specific option descriptions and usage.

• See the Red Hat Ceph Storage Architecture Guide for details about using on-wire encryption with the Ceph messenger version 2 protocol.

### 2.4. Configuring a Public Network

To configure Ceph networks, you must add a network configuration to the [global] section of the configuration file. Note that the IP addresses you set in your network configuration are different from the public-facing IP addresses that network clients may use to access your service.

Ceph functions perfectly well with only a public network. However, Ceph allows you to establish much more specific criteria, including multiple IP networks for your public network.

You can also establish a separate, private cluster network to handle OSD heartbeat, object replication, and recovery traffic. For more information about the private network, see Configuring a private network.

**NOTE**

Ceph uses CIDR notation for subnets, for example, 10.0.0.0/24. Typical internal IP networks are often 192.168.0.0/24 or 10.0.0.0/24.

**NOTE**

If you specify more than one IP address for either the public or the cluster network, the subnets within the network must be capable of routing to each other. In addition, make sure you include each IP address in your IP tables, and open ports for them as necessary.

The public network configuration allows you specifically define IP addresses and subnets for the public network. You may specifically assign static IP addresses or use the public_addr setting for a specific daemon to override public network settings.

**Prerequisites**

- Installation of the Red Hat Ceph Storage software.

**Procedure**

1. Add the following option to the [global] section of the Ceph configuration file. You may specify multiple subnets by separating the CIDR formatted IP addresses with commas.

   **Syntax**

   ```
   [global]
   ...
   public_network = IP-ADDRESS
   ```

2. To set the public network for a specific daemon, add the public_addr option to the section for that daemon in the Ceph configuration file:

   **Syntax**

   ```
   [daemon]
   ...
   public_addr = IP-ADDRESS
   ```
public_addr = IP-ADDRESS

Example

```
[mon]
...
public_addr = 192.168.0.0/24
```

2. Restart the cluster or restart the daemon. Ceph daemons bind dynamically, so you do not have to restart the entire cluster at once if you change the network configuration for a specific daemon.

Additional Resources

- See the common options in *Red Hat Ceph Storage Configuration Guide, Appendix B* for specific option descriptions and usage.

### 2.5. CONFIGURING A PRIVATE NETWORK

Network configuration settings are not required. Ceph assumes a public network with all hosts operating on it, unless you specifically configure a cluster network, also known as a *private network*.

If you create a cluster network, OSDs will route heartbeat, object replication, and recovery traffic over the cluster network. This can improve performance, compared to using a single network.

**IMPORTANT**

For added security, the cluster network should not be reachable from the public network or the Internet.

To assign a cluster network, use the `--cluster-network` option with the `cephadm bootstrap` command. The cluster network that you specify must define a subnet in CIDR notation (for example, 10.90.90.0/24 or fe80::/64).

**Prerequisites**

- Access to the Ceph software repository.
- Root-level access to all nodes in the storage cluster.

**Procedure**

1. Run the `cephadm bootstrap` command from the initial node that you want to use as the Monitor node in the storage cluster. Include the `--cluster-network` option in the command.

**Syntax**

```
cephadm bootstrap --mon-ip IP-ADDRESS --registry-url registry.redhat.io --registry-username USER_NAME --registry-password PASSWORD --cluster-network NETWORK-IP-ADDRESS
```

**Example**
Additional Resources

- For more information about invoking `cephadm bootstrap`, see the Bootstrapping a new storage cluster section in the Red Hat Ceph Storage Installation Guide.

### 2.6. VERIFY THE FIREWALL SETTINGS

By default, daemons bind to ports within the `6800:7100` range. You can configure this range at your discretion. Before configuring the firewall, check the default firewall configuration.

#### Prerequisites

- A running Red Hat Ceph Storage cluster.
- Access to the Ceph software repository.
- Root-level access to the Ceph Monitor node.

#### Procedure

1. You can configure this range at your discretion:
   
   ```
   [root@mon ~]# sudo iptables -L
   ```

2. For the `firewalld` daemon, execute the following command:
   
   ```
   [root@mon ~]# firewall-cmd --list-all-zones
   ```

   Some Linux distributions include rules that reject all inbound requests except SSH from all network interfaces.

   **Example**

   ```
   REJECT all -- anywhere anywhere reject-with icmp-host-prohibited
   ```

### 2.7. FIREWALL SETTINGS FOR CEPH MONITOR NODE

You can enable encryption for all Ceph traffic over the network with the introduction of the messenger version 2 protocol. The `secure` mode setting for messenger v2 encrypts communication between Ceph daemons and Ceph clients, giving you end-to-end encryption.

#### Messenger v2 Protocol

The second version of Ceph’s on-wire protocol, `msgr2`, includes several new features:

- A secure mode encrypts all data moving through the network.
- Encapsulation improvement of authentication payloads.
- Improvements to feature advertisement and negotiation.
The Ceph daemons bind to multiple ports allowing both the legacy, v1-compatible, and the new, v2-compatible, Ceph clients to connect to the same storage cluster. Ceph clients or other Ceph daemons connecting to the Ceph Monitor daemon will try to use the v2 protocol first, if possible, but if not, then the legacy v1 protocol will be used. By default, both messenger protocols, v1 and v2, are enabled. The new v2 port is 3300, and the legacy v1 port is 6789, by default.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Access to the Ceph software repository.
- Root-level access to the Ceph Monitor node.

Procedure

1. Add rules using the following example:

   ```bash
   [root@mon ~]# sudo iptables -A INPUT -i IFACE -p tcp -s IP-ADDRESS/NETMASK --dport 6789 -j ACCEPT
   [root@mon ~]# sudo iptables -A INPUT -i IFACE -p tcp -s IP-ADDRESS/NETMASK --dport 3300 -j ACCEPT
   ``

   a. Replace IFACE with the public network interface (for example, eth0, eth1, and so on).

   b. Replace IP-ADDRESS with the IP address of the public network and NETMASK with the netmask for the public network.

2. For the firewalld daemon, execute the following commands:

   ```bash
   [root@mon ~]# firewall-cmd --zone=public --add-port=6789/tcp
   [root@mon ~]# firewall-cmd --zone=public --add-port=6789/tcp --permanent
   [root@mon ~]# firewall-cmd --zone=public --add-port=3300/tcp
   [root@mon ~]# firewall-cmd --zone=public --add-port=3300/tcp --permanent
   ```

2.8. FIREWALL SETTINGS FOR CEPH OSDS

By default, Ceph OSDs bind to the first available ports on a Ceph node beginning at port 6800. Ensure to open at least four ports beginning at port 6800 for each OSD that runs on the node:

- One for talking to clients and monitors on the public network.
- One for sending data to other OSDs on the cluster network.
- Two for sending heartbeat packets on the cluster network.
Ports are node-specific. However, you might need to open more ports than the number of ports needed by Ceph daemons running on that Ceph node in the event that processes get restarted and the bound ports do not get released. Consider opening a few additional ports in case a daemon fails and restarts without releasing the port such that the restarted daemon binds to a new port. Also, consider opening the port range of `6800:7300` on each OSD node.

If you set separate public and cluster networks, you must add rules for both the public network and the cluster network, because clients will connect using the public network and other Ceph OSD Daemons will connect using the cluster network.

**Prerequisites**

- A running Red Hat Ceph Storage cluster.
- Access to the Ceph software repository.
- Root-level access to the Ceph OSD nodes.

**Procedure**

1. Add rules using the following example:

   ```bash
   [root@mon ~]# sudo iptables -A INPUT -i IFACE -m multiport -p tcp -s IP-ADDRESS/NETMASK --dports 6800:6810 -j ACCEPT
   ``

   a. Replace `IFACE` with the public network interface (for example, `eth0`, `eth1`, and so on).

   b. Replace `IP-ADDRESS` with the IP address of the public network and `NETMASK` with the netmask for the public network.

2. For the `firewalld` daemon, execute the following:

   ```bash
   [root@mon ~]# firewall-cmd --zone=public --add-port=6800-6810/tcp
   [root@mon ~]# firewall-cmd --zone=public --add-port=6800-6810/tcp --permanent
   ``

   If you put the cluster network into another zone, open the ports within that zone as appropriate.

**2.9. ADDITIONAL RESOURCES**

- See the Red Hat Ceph Storage network configuration options in Appendix B for specific option descriptions and usage.
See the Red Hat Ceph Storage Architecture Guide for details about using on-wire encryption with the Ceph messenger version 2 protocol.
CHAPTER 3. CEPH MONITOR CONFIGURATION

As a storage administrator, you can use the default configuration values for the Ceph Monitor or customize them according to the intended workload.

3.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

3.2. CEPH MONITOR CONFIGURATION

Understanding how to configure a Ceph Monitor is an important part of building a reliable Red Hat Ceph Storage cluster. All storage clusters have at least one monitor. A Ceph Monitor configuration usually remains fairly consistent, but you can add, remove or replace a Ceph Monitor in a storage cluster.

Ceph monitors maintain a "master copy" of the cluster map. That means a Ceph client can determine the location of all Ceph monitors and Ceph OSDs just by connecting to one Ceph monitor and retrieving a current cluster map.

Before Ceph clients can read from or write to Ceph OSDs, they must connect to a Ceph Monitor first. With a current copy of the cluster map and the CRUSH algorithm, a Ceph client can compute the location for any object. The ability to compute object locations allows a Ceph client to talk directly to Ceph OSDs, which is a very important aspect of Ceph's high scalability and performance.

The primary role of the Ceph Monitor is to maintain a master copy of the cluster map. Ceph Monitors also provide authentication and logging services. Ceph Monitors write all changes in the monitor services to a single Paxos instance, and Paxos writes the changes to a key-value store for strong consistency. Ceph Monitors can query the most recent version of the cluster map during synchronization operations. Ceph Monitors leverage the key-value store’s snapshots and iterators, using the rocksdb database, to perform store-wide synchronization.

3.2.1. Accessing the configuration database

![Diagram of Ceph Monitor Configuration](image)
Red Hat Ceph Storage centralizes configuration within the Ceph Monitors instead of using the Ceph configuration file `/etc/ceph.conf`. `ceph.conf` has been deprecated for Red Hat Ceph Storage 5.

The configuration database that the Ceph Monitors use has the same semantic structure as `ceph.conf`. Any changes to the configuration are applied to daemons or clients in the system immediately. Restarting the clients is no longer needed.

When a client or daemon starts, it can still use a local `ceph.conf` or DNS to find the Ceph Monitor addresses, but the usual configuration first reads its settings from the Ceph Monitors.

**Prerequisites**

- A running storage cluster.
- Root-level access to a Ceph Monitor node.

**Procedure**

1. Use the `ceph config` command to access the configuration database:

   **Example**
   ```
   [root@mon ~]# cephadm shell ceph config
   ```

2. Edit the settings in the `[mon]` section of the file.

3. Save and exit the file.

**Additional Resources**

- For more information about the options available for the `ceph config` command, use `ceph config -h`.

### 3.3. CEPH CLUSTER MAPS

The cluster map is a composite of maps, including the monitor map, the OSD map, and the placement group map. The cluster map tracks a number of important events:

- Which processes are in the Red Hat Ceph Storage cluster.
- Which processes that are in the Red Hat Ceph Storage cluster are up and running or down.
- Whether, the placement groups are active or inactive, and clean or in some other state.
- other details that reflect the current state of the cluster such as:
  - the total amount of storage space or
  - the amount of storage used.

When there is a significant change in the state of the cluster, for example, a Ceph OSD goes down, a placement group falls into a degraded state, and so on. The cluster map gets updated to reflect the current state of the cluster. Additionally, the Ceph monitor also maintains a history of the prior states of the cluster. The monitor map, OSD map, and placement group map each maintain a history of their map versions. Each version is called an epoch.
When operating the Red Hat Ceph Storage cluster, keeping track of these states is an important part of the cluster administration.

3.4. CEPH MONITOR QUORUM

A cluster will run sufficiently with a single monitor. However, a single monitor is a single-point-of-failure. To ensure high availability in a production Ceph storage cluster, run Ceph with multiple monitors so that the failure of a single monitor will not cause a failure of the entire storage cluster.

When a Ceph storage cluster runs multiple Ceph Monitors for high availability, Ceph Monitors use the Paxos algorithm to establish consensus about the master cluster map. A consensus requires a majority of monitors running to establish a quorum for consensus about the cluster map. For example, 1; 2 out of 3; 3 out of 5; 4 out of 6; and so on.

Red Hat recommends running a production Red Hat Ceph Storage cluster with at least three Ceph Monitors to ensure high availability. When you run multiple monitors, you can specify the initial monitors that must be members of the storage cluster to establish a quorum. This may reduce the time it takes for the storage cluster to come online.

```
[mon]
mon_initial_members = a,b,c
```

NOTE

A majority of the monitors in the storage cluster must be able to reach each other in to establish a quorum. You can decrease the initial number of monitors to establish a quorum with the `mon_initial_members` option.

3.5. CEPH MONITOR CONSISTENCY

When you add monitor settings to the Ceph configuration file, you need to be aware of some of the architectural aspects of Ceph Monitors. Ceph imposes strict consistency requirements for a Ceph Monitor when discovering another Ceph Monitor within the cluster. Whereas Ceph clients and other Ceph daemons use the Ceph configuration file to discover monitors, monitors discover each other using the monitor map (`monmap`), not the Ceph configuration file.

A Ceph Monitor always refers to the local copy of the monitor map when discovering other Ceph Monitors in the Red Hat Ceph Storage cluster. Using the monitor map instead of the Ceph configuration file avoids errors that could break the cluster. For example, typos in the Ceph configuration file when specifying a monitor address or port. Since monitors use monitor maps for discovery and they share monitor maps with clients and other Ceph daemons, the monitor map provides monitors with a strict guarantee that their consensus is valid.

Strict consistency when applying updates to the monitor maps

As with any other updates on the Ceph Monitor, changes to the monitor map always run through a distributed consensus algorithm called Paxos. The Ceph Monitors must agree on each update to the monitor map, such as adding or removing a Ceph Monitor, to ensure that each monitor in the quorum has the same version of the monitor map. Updates to the monitor map are incremental so that Ceph Monitors have the latest agreed-upon version and a set of previous versions.

Maintaining history

Maintaining a history enables a Ceph Monitor that has an older version of the monitor map to catch up with the current state of the Red Hat Ceph Storage cluster.
If Ceph Monitors discovered each other through the Ceph configuration file instead of through the monitor map, it would introduce additional risks because the Ceph configuration files are not updated and distributed automatically. Ceph Monitors might inadvertently use an older Ceph configuration file, fail to recognize a Ceph Monitor, fall out of a quorum, or develop a situation where Paxos is not able to determine the current state of the system accurately.

### 3.6. Bootstrap the Ceph Monitor

In most configuration and deployment cases, tools that deploy Ceph, such as `cephadm`, might help bootstrap the Ceph monitors by generating a monitor map for you.

A Ceph monitor requires a few explicit settings:

- **File System ID**: The `fsid` is the unique identifier for your object store. Since you can run multiple storage clusters on the same hardware, you must specify the unique ID of the object store when bootstrapping a monitor. Using deployment tools, such as `cephadm`, will generate a file system identifier, but you can also specify the `fsid` manually.

- **Monitor ID**: A monitor ID is a unique ID assigned to each monitor within the cluster. It is an alphanumeric value, and by convention, the identifier usually follows an alphabetical increment. For example, `a`, `b`, and so on. This can be set in the Ceph configuration file. For example, `[mon.a]`, `[mon.b]`, and so on, by a deployment tool, or using the `ceph` command.

- **Keys**: The monitor must have secret keys.

**Additional Resources**

- For more information about `cephadm` and the Ceph orchestrator, see the *Red Hat Ceph Storage Operations Guide*.

### 3.7. Ceph Monitor Section in the Configuration File

To apply configuration settings to the entire cluster, enter the configuration settings under the `[global]` section. To apply configuration settings to all monitors in the cluster, enter the configuration settings under the `[mon]` section. To apply configuration settings to specific monitors, specify the monitor instance.

**Example**

`[mon.a]`

By convention, monitor instance names use alpha notation.

```
[global]
[mon]
[mon.a]
[mon.b]
[mon.c]
```

### 3.8. Minimum Configuration for a Ceph Monitor
The bare minimum monitor settings for a Ceph Monitor in the Ceph configuration file includes a host name for each monitor if it is not configured for DNS and the monitor address. The Ceph Monitors run on port 6789 and 3300 by default.

**IMPORTANT**

Do not edit the Ceph configuration file.

**NOTE**

This minimum configuration for monitors assumes that a deployment tool generates the fsid and the mon. key for you.

You can use the following commands to set or read the storage cluster configuration options.

- `ceph config dump` - Dumps the entire configuration database for the whole storage cluster.
- `ceph config generate-minimal-conf` - Generates a minimal ceph.conf file.
- `ceph config get WHO` - Dumps the configuration for a specific daemon or a client, as stored in the Ceph Monitor’s configuration database.
- `ceph config set WHO OPTION VALUE` - Sets the configuration option in the Ceph Monitor’s configuration database.
- `ceph config show WHO` - Shows the reported running configuration for a running daemon.
- `ceph config assimilate-conf -i INPUT_FILE -o OUTPUT_FILE` - Ingests a configuration file from the input file and moves any valid options into the Ceph Monitors’ configuration database.

Here, WHO parameter might be name of the section or a Ceph daemon, OPTION is a configuration file, and VALUE can be either true or false.

**IMPORTANT**

When a Ceph daemon needs a config option prior to getting the option from the config store, you can set the configuration by running the following command:

```
ceph cephadm set-extra-ceph-conf
```

This command adds text to all the daemon’s ceph.conf files. It is a workaround and is NOT a recommended operation.

### 3.9. UNIQUE IDENTIFIER FOR CEPH

Each Red Hat Ceph Storage cluster has a unique identifier (fsid). If specified, it usually appears under the [global] section of the configuration file. Deployment tools usually generate the fsid and store it in the monitor map, so the value may not appear in a configuration file. The fsid makes it possible to run daemons for multiple clusters on the same hardware.

**NOTE**

Do not set this value if you use a deployment tool that does it for you.
3.10. CEPH MONITOR DATA STORE

Ceph provides a default path where Ceph monitors store data.

**IMPORTANT**

Red Hat recommends running Ceph monitors on separate hosts and drives from Ceph OSDs for optimal performance in a production Red Hat Ceph Storage cluster.

Ceph monitors call the \texttt{fsync()} function often, which can interfere with Ceph OSD workloads.

Ceph monitors store their data as key-value pairs. Using a data store prevents recovering Ceph monitors from running corrupted versions through Paxos, and it enables multiple modification operations in one single atomic batch, among other advantages.

**IMPORTANT**

Red Hat does not recommend changing the default data location. If you modify the default location, make it uniform across Ceph monitors by setting it in the \texttt{[mon]} section of the configuration file.

3.11. CEPH STORAGE CAPACITY

When a Red Hat Ceph Storage cluster gets close to its maximum capacity (specifies by the \texttt{mon OSD full ratio} parameter), Ceph prevents you from writing to or reading from Ceph OSDs as a safety measure to prevent data loss. Therefore, letting a production Red Hat Ceph Storage cluster approach its full ratio is not a good practice, because it sacrifices high availability. The default full ratio is \texttt{.95}, or 95% of capacity. This a very aggressive setting for a test cluster with a small number of OSDs.

**TIP**

When monitoring a cluster, be alert to warnings related to the \texttt{nearfull} ratio. This means that a failure of some OSDs could result in a temporary service disruption if one or more OSDs fails. Consider adding more OSDs to increase storage capacity.

A common scenario for test clusters involves a system administrator removing a Ceph OSD from the Red Hat Ceph Storage cluster to watch the cluster re-balance. Then, removing another Ceph OSD, and so on until the Red Hat Ceph Storage cluster eventually reaches the full ratio and locks up.

**IMPORTANT**

Red Hat recommends a bit of capacity planning even with a test cluster. Planning enables you to gauge how much spare capacity you will need in to maintain high availability.

Ideally, you want to plan for a series of Ceph OSD failures where the cluster can recover to an \texttt{active + clean} state without replacing those Ceph OSDs immediately. You can run a cluster in an \texttt{active + degraded} state, but this is not ideal for normal operating conditions.

The following diagram depicts a simplistic Red Hat Ceph Storage cluster containing 33 Ceph Nodes with one Ceph OSD per host, each Ceph OSD Daemon reading from and writing to a 3TB drive. So this exemplary Red Hat Ceph Storage cluster has a maximum actual capacity of 99TB. With a \texttt{mon OSD full}
If the Red Hat Ceph Storage cluster falls to 5 TB of remaining capacity, the cluster will not allow Ceph clients to read and write data. So the Red Hat Ceph Storage cluster’s operating capacity is 95 TB, not 99 TB.

Identify two numbers for your cluster:

- the number of OSDs
- the total capacity of the cluster

To determine the mean average capacity of an OSD within a cluster, divide the total capacity of the cluster by the number of OSDs in the cluster. Consider multiplying that number by the number of OSDs you expect to fail simultaneously during normal operations (a relatively small number). Finally, multiply the capacity of the cluster by the full ratio to arrive at a maximum operating capacity. Then, subtract the amount of data from the OSDs you expect to fail to arrive at a reasonable full ratio. Repeat the foregoing process with a higher number of OSD failures (for example, a rack of OSDs) to arrive at a reasonable number for a near full ratio.

### 3.12. CEPH HEARTBEAT

Ceph monitors know about the cluster by requiring reports from each OSD, and by receiving reports from OSDs about the status of their neighboring OSDs. Ceph provides reasonable default settings for interaction between monitor and OSD, however, you can modify them as needed.

### 3.13. CEPH MONITOR SYNCHRONIZATION ROLE
When you run a production cluster with multiple monitors which is recommended, each monitor checks to see if a neighboring monitor has a more recent version of the cluster map. For example, a map in a neighboring monitor with one or more epoch numbers higher than the most current epoch in the map of the instant monitor. Periodically, one monitor in the cluster might fall behind the other monitors to the point where it must leave the quorum, synchronize to retrieve the most current information about the cluster, and then rejoin the quorum.

**Synchronization roles**

For the purposes of synchronization, monitors can assume one of three roles:

- **Leader:** The Leader is the first monitor to achieve the most recent Paxos version of the cluster map.
- **Provider:** The Provider is a monitor that has the most recent version of the cluster map, but was not the first to achieve the most recent version.
- **Requester:** The Requester is a monitor that has fallen behind the leader and must synchronize to retrieve the most recent information about the cluster before it can rejoin the quorum.

These roles enable a leader to delegate synchronization duties to a provider, which prevents synchronization requests from overloading the leader and improving performance. In the following diagram, the requester has learned that it has fallen behind the other monitors. The requester asks the leader to synchronize, and the leader tells the requester to synchronize with a provider.

**Monitor synchronization**

Synchronization always occurs when a new monitor joins the cluster. During runtime operations, monitors can receive updates to the cluster map at different times. This means the leader and provider roles may migrate from one monitor to another. If this happens while synchronizing, for example, a provider falls behind the leader, the provider can terminate synchronization with a requester.

Once synchronization is complete, Ceph requires trimming across the cluster. Trimming requires that the placement groups are **active + clean**.

3.14. CEPH TIME SYNCHRONIZATION
Ceph daemons pass critical messages to each other, which must be processed before daemons reach a timeout threshold. If the clocks in Ceph monitors are not synchronized, it can lead to a number of anomalies.

For example:

- Daemons ignoring received messages such as outdated timestamps.
- Timeouts triggered too soon or late when a message was not received in time.

**TIP**

Install NTP on the Ceph monitor hosts to ensure that the monitor cluster operates with synchronized clocks.

Clock drift may still be noticeable with NTP even though the discrepancy is not yet harmful. Ceph clock drift and clock skew warnings can get triggered even though NTP maintains a reasonable level of synchronization. Increasing your clock drift may be tolerable under such circumstances. However, a number of factors such as workload, network latency, configuring overrides to default timeouts, and other synchronization options can influence the level of acceptable clock drift without compromising Paxos guarantees.

**Additional Resources**

- See the section on *Ceph time synchronization* for more details.

**3.15. ADDITIONAL RESOURCES**

- See all the Red Hat Ceph Storage Monitor configuration options in *Appendix C* for specific option descriptions and usage.
CHAPTER 4. CEPH AUTHENTICATION CONFIGURATION

As a storage administrator, authenticating users and services is important to the security of the Red Hat Ceph Storage cluster. Red Hat Ceph Storage includes the Cephx protocol, as the default, for cryptographic authentication, and the tools to manage authentication in the storage cluster.

4.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

4.2. CEPHX AUTHENTICATION

The cephx protocol is enabled by default. Cryptographic authentication has some computational costs, though they are generally quite low. If the network environment connecting clients and hosts is considered safe and you cannot afford authentication computational costs, you can disable it. When deploying a Ceph storage cluster, the deployment tool will create the client.admin user and keyring.

IMPORTANT

Red Hat recommends using authentication.

NOTE

If you disable authentication, you are at risk of a man-in-the-middle attack altering client and server messages, which could lead to significant security issues.

Enabling and disabling Cephx

Enabling Cephx requires that you have deployed keys for the Ceph Monitors and OSDs. When toggling Cephx authentication on or off, you do not have to repeat the deployment procedures.

4.3. ENABLING CEPHX

When cephx is enabled, Ceph will look for the keyring in the default search path, which includes /etc/ceph/$cluster.$name.keyring. You can override this location by adding a keyring option in the [global] section of the Ceph configuration file, but this is not recommended.

Execute the following procedures to enable cephx on a cluster with authentication disabled. If you or your deployment utility have already generated the keys, you may skip the steps related to generating keys.

Prerequisites

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor node.

Procedure

1. Create a client.admin key, and save a copy of the key for your client host:

   [root@mon ~]# ceph auth get-or-create client.admin mon 'allow *' osd 'allow *' -o /etc/ceph/ceph.client.admin.keyring
2. Create a keyring for the monitor cluster and generate a monitor secret key:

   [root@mon ~]# ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *

3. Copy the monitor keyring into a `ceph.mon.keyring` file in every monitor `mon data` directory. For example, to copy it to `mon.a` in cluster `ceph`, use the following:

   [root@mon ~]# cp /tmp/ceph.mon.keyring /var/lib/ceph/mon/ceph-a/keyring

4. Generate a secret key for every OSD, where `ID` is the OSD number:

   ceph auth get-or-create osd.\Id mon 'allow rwx' osd 'allow *' -o /var/lib/ceph/osd/ceph-\Id/keyring

5. By default the `cephx` authentication protocol is enabled.

   **NOTE**
   
   If the `cephx` authentication protocol was disabled previously by setting the authentication options to `none`, then by removing the following lines under the `[global]` section in the Ceph configuration file (`/etc/ceph/ceph.conf`) will reenable the `cephx` authentication protocol:

   ```
   auth_cluster_required = none
   auth_service_required = none
   auth_client_required = none
   ```

6. Start or restart the Ceph storage cluster.
Enabling *cephx* requires downtime because the cluster needs to be completely restarted, or it needs to be shut down and then started while client I/O is disabled.

These flags need to be set before restarting or shutting down the storage cluster:

```
[root@mon ~]# ceph osd set noout
[root@mon ~]# ceph osd set norecover
[root@mon ~]# ceph osd set norebalance
[root@mon ~]# ceph osd set nobackfill
[root@mon ~]# ceph osd set nodown
[root@mon ~]# ceph osd set pause
```

Once *cephx* is enabled and all PGs are active and clean, unset the flags:

```
[root@mon ~]# ceph osd unset noout
[root@mon ~]# ceph osd unset norecover
[root@mon ~]# ceph osd unset norebalance
[root@mon ~]# ceph osd unset nobackfill
[root@mon ~]# ceph osd unset nodown
[root@mon ~]# ceph osd unset pause
```

### 4.4. DISABLING CEPHX

The following procedure describes how to disable Cephx. If your cluster environment is relatively safe, you can offset the computation expense of running authentication.

**IMPORTANT**

Red Hat recommends enabling authentication.

However, it may be easier during setup or troubleshooting to temporarily disable authentication.

**Prerequisites**

- A running Red Hat Ceph Storage cluster.
- Root-level access to the Ceph Monitor node.

**Procedure**

1. Disable *cephx* authentication by setting the following options in the `[global]` section of the Ceph configuration file:

   **Example**

   ```
   auth_cluster_required = none
   auth_service_required = none
   auth_client_required = none
   ```

2. Start or restart the Ceph storage cluster.
4.5. CEPHX USER KEYRINGS

When you run Ceph with authentication enabled, the `ceph` administrative commands and Ceph clients require authentication keys to access the Ceph storage cluster.

The most common way to provide these keys to the `ceph` administrative commands and clients is to include a Ceph keyring under the `/etc/ceph/` directory. The file name is usually `ceph.client.admin.keyring` or `$cluster.client.admin.keyring`. If you include the keyring under the `/etc/ceph/` directory, you do not need to specify a `keyring` entry in the Ceph configuration file.

**IMPORTANT**

Red Hat recommends copying the Red Hat Ceph Storage cluster keyring file to nodes where you will run administrative commands, because it contains the `client.admin` key.

To do so, execute the following command:

```
# scp USER@HOSTNAME:/etc/ceph/ceph.client.admin.keyring /etc/ceph/ceph.client.admin.keyring
```

Replace `USER` with the user name used on the host with the `client.admin` key and `HOSTNAME` with the host name of that host.

**NOTE**

Ensure the `ceph.keyring` file has appropriate permissions set on the client machine.

You can specify the key itself in the Ceph configuration file using the `key` setting, which is not recommended, or a path to a key file using the `keyfile` setting.

4.6. CEPHX DAEMON KEYRINGS

Administrative users or deployment tools might generate daemon keyrings in the same way as generating user keyrings. By default, Ceph stores daemons keyrings inside their data directory. The default keyring locations, and the capabilities necessary for the daemon to function.

**NOTE**

The monitor keyring contains a key but no capabilities, and is not part of the Ceph storage cluster `auth` database.

The daemon data directory locations default to directories of the form:

```
/var/lib/ceph/$type/CLUSTER-ID
```

Example

```
/var/lib/ceph/osd/ceph-12
```

You can override these locations, but it is not recommended.

4.7. CEPHX MESSAGE SIGNATURES
Ceph provides fine-grained control so you can enable or disable signatures for service messages between the client and Ceph. You can enable or disable signatures for messages between Ceph daemons.

**IMPORTANT**

Red Hat recommends that Ceph authenticate all ongoing messages between the entities using the session key set up for that initial authentication.

**NOTE**

Ceph kernel modules do not support signatures yet.

### 4.8. ADDITIONAL RESOURCES

- See all the Red Hat Ceph Storage Cephx configuration options in *Appendix D* for specific option descriptions and usage.
CHAPTER 5. POOLS, PLACEMENT GROUPS, AND CRUSH CONFIGURATION

As a storage administrator, you can choose to use the Red Hat Ceph Storage default options for pools, placement groups, and the CRUSH algorithm or customize them for the intended workload.

5.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

5.2. POOLS PLACEMENT GROUPS AND CRUSH

When you create pools and set the number of placement groups for the pool, Ceph uses default values when you do not specifically override the defaults.

**IMPORTANT**

Red Hat recommends overriding some of the defaults. Specifically, set a pool’s replica size and override the default number of placement groups.

You can set these values when running pool commands. You can also override the defaults by adding new ones in the [global] section of the Ceph configuration file.

**Example**

```bash
[global]

# By default, Ceph makes 3 replicas of objects. If you want to set 4
# copies of an object as the default value—a primary copy and three replica
# copies—reset the default values as shown in 'osd pool default size'.
# If you want to allow Ceph to write a lesser number of copies in a degraded
# state, set 'osd pool default min size' to a number less than the
# 'osd pool default size' value.

osd_pool_default_size = 4  # Write an object 4 times.
osd_pool_default_min_size = 1 # Allow writing one copy in a degraded state.

# Ensure you have a realistic number of placement groups. We recommend
# approximately 100 per OSD. E.g., total number of OSDs multiplied by 100
# divided by the number of replicas (i.e., osd pool default size). So for
# 10 OSDs and osd pool default size = 4, we'd recommend approximately
# (100 * 10) / 4 = 250.

osd_pool_default_pg_num = 250
osd_pool_default_pgp_num = 250
```

5.3. ADDITIONAL RESOURCES

- See all the Red Hat Ceph Storage pool, placement group, and CRUSH configuration options in Appendix E for specific option descriptions and usage.
CHAPTER 6. CEPH OBJECT STORAGE DAEMON (OSD) CONFIGURATION

As a storage administrator, you can configure the Ceph Object Storage Daemon (OSD) to be redundant and optimized based on the intended workload.

6.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

6.2. CEPH OSD CONFIGURATION

All Ceph clusters have a configuration, which defines:

- Cluster identity
- Authentication settings
- Ceph daemon membership in the cluster
- Network configuration
- Host names and addresses
- Paths to keyrings
- Paths to OSD log files
- Other runtime options

A deployment tool, such as `cephadm`, will typically create an initial Ceph configuration file for you. However, you can create one yourself if you prefer to bootstrap a cluster without using a deployment tool.

For your convenience, each daemon has a series of default values. Many are set by the `ceph/src/common/config_opts.h` script. You can override these settings with a Ceph configuration file or at runtime by using the monitor `tell` command or connecting directly to a daemon socket on a Ceph node.

**IMPORTANT**

Red Hat does not recommend changing the default paths, as it makes it more difficult to troubleshoot Ceph later.

Additional Resources


6.3. SCRUBBING THE OSD

In addition to making multiple copies of objects, Ceph ensures data integrity by scrubbing placement groups. Ceph scrubbing is analogous to the `fsck` command on the object storage layer.
For each placement group, Ceph generates a catalog of all objects and compares each primary object and its replicas to ensure that no objects are missing or mismatched.

Light scrubbing (daily) checks the object size and attributes. Deep scrubbing (weekly) reads the data and uses checksums to ensure data integrity.

Scrubbing is important for maintaining data integrity, but it can reduce performance. Adjust the following settings to increase or decrease scrubbing operations.

**Additional resources**

- See [Ceph scrubbing options](#) in the appendix of the *Red Hat Ceph Storage Configuration Guide* for more details.

### 6.4. BACKFILLING AN OSD

When you add Ceph OSDs to a cluster or remove them from the cluster, the CRUSH algorithm rebalances the cluster by moving placement groups to or from Ceph OSDs to restore the balance. The process of migrating placement groups and the objects they contain can reduce the cluster operational performance considerably. To maintain operational performance, Ceph performs this migration with the 'backfill' process, which allows Ceph to set backfill operations to a lower priority than requests to read or write data.

### 6.5. OSD RECOVERY

When the cluster starts or when a Ceph OSD terminates unexpectedly and restarts, the OSD begins peering with other Ceph OSDs before a write operation can occur.

If a Ceph OSD crashes and comes back online, usually it will be out of sync with other Ceph OSDs containing more recent versions of objects in the placement groups. When this happens, the Ceph OSD goes into recovery mode and seeks to get the latest copy of the data and bring its map back up to date. Depending upon how long the Ceph OSD was down, the OSD's objects and placement groups may be significantly out of date. Also, if a failure domain went down, for example, a rack, more than one Ceph OSD might come back online at the same time. This can make the recovery process time consuming and resource intensive.

To maintain operational performance, Ceph performs recovery with limitations on the number of recovery requests, threads, and object chunk sizes which allows Ceph to perform well in a degraded state.

### 6.6. ADDITIONAL RESOURCES

- See all the Red Hat Ceph Storage Ceph OSD configuration options in *Appendix F* for specific option descriptions and usage.
CHAPTER 7. CEPH MONITOR AND OSD INTERACTION CONFIGURATION

As a storage administrator, you must properly configure the interactions between the Ceph Monitors and OSDs to ensure a stable working environment.

7.1. PREREQUISITES

- Installation of the Red Hat Ceph Storage software.

7.2. CEPH MONITOR AND OSD INTERACTION

After you have completed your initial Ceph configuration, you can deploy and run Ceph. When you execute a command such as ceph health or ceph -s, the Ceph Monitor reports on the current state of the Ceph storage cluster. The Ceph Monitor knows about the Ceph storage cluster by requiring reports from each Ceph OSD daemon, and by receiving reports from Ceph OSD daemons about the status of their neighboring Ceph OSD daemons. If the Ceph Monitor does not receive reports, or if it receives reports of changes in the Ceph storage cluster, the Ceph Monitor updates the status of the Ceph cluster map.

Ceph provides reasonable default settings for Ceph Monitor and OSD interaction. However, you can override the defaults. The following sections describe how Ceph Monitors and Ceph OSD daemons interact for the purposes of monitoring the Ceph storage cluster.

7.3. OSD HEARTBEAT

Each Ceph OSD daemon checks the heartbeat of other Ceph OSD daemons every 6 seconds. To change the heartbeat interval, change the value at runtime:

Syntax

```
ceph config set osd osd_heartbeat_interval TIME_IN_SECONDS
```

Example

```
[ceph: root@host01 /]# ceph config set osd osd_heartbeat_interval 60
```

If a neighboring Ceph OSD daemon does not send heartbeat packets within a 20 second grace period, the Ceph OSD daemon might consider the neighboring Ceph OSD daemon down. It can report it back to a Ceph Monitor, which updates the Ceph cluster map. To change the grace period, set the value at runtime:

Syntax

```
ceph config set osd osd_heartbeat_grace TIME_IN_SECONDS
```

Example

```
[ceph: root@host01 /]# ceph config set osd osd_heartbeat_grace 30
```
7.4. REPORTING AN OSD AS DOWN

By default, two Ceph OSD Daemons from different hosts must report to the Ceph Monitors that another Ceph OSD Daemon is down before the Ceph Monitors acknowledge that the reported Ceph OSD Daemon is down.

However, there is the chance that all the OSDs reporting the failure are in different hosts in a rack with a bad switch that causes connection problems between OSDs.

To avoid a “false alarm,” Ceph considers the peers reporting the failure as a proxy for a “subcluster” that is similarly laggy. While this is not always the case, it may help administrators localize the grace correction to a subset of the system that is performing poorly.

Ceph uses the mon_osd_reporter_subtree_level setting to group the peers into the “subcluster” by their common ancestor type in the CRUSH map.

By default, only two reports from a different subtree are required to report another Ceph OSD Daemon down. Administrators can change the number of reporters from unique subtrees and the common ancestor type required to report a Ceph OSD Daemon down to a Ceph Monitor by setting the mon_osd_min_down_reporters and mon_osd_reporter_subtree_level values at runtime:

Syntax

```plaintext
ceph config set mon mon_osd_min_down_reporters NUMBER
```

Example

```plaintext
[ceph: root@host01 ]# ceph config set mon mon_osd_min_down_reporters 4
```

Syntax
ceph config set mon mon_osd_reporter_subtree_level CRUSH_ITEM

Example

[ceph: root@host01 /]# ceph config set mon mon_osd_reporter_subtree_level host
[ceph: root@host01 /]# ceph config set mon mon_osd_reporter_subtree_level rack
[ceph: root@host01 /]# ceph config set mon mon_osd_reporter_subtree_level osd

7.5. REPORTING A PEERING FAILURE

If a Ceph OSD daemon cannot peer with any of the Ceph OSD daemons defined in its Ceph configuration file or the cluster map, it pings a Ceph Monitor for the most recent copy of the cluster map every 30 seconds. You can change the Ceph Monitor heartbeat interval by setting the value at runtime:

Syntax

ceph config set osd osd_mon_heartbeat_interval TIME_IN_SECONDS

Example

[ceph: root@host01 /]# ceph config set osd osd_mon_heartbeat_interval 60
7.6. OSD REPORTING STATUS

If a Ceph OSD Daemon does not report to a Ceph Monitor, the Ceph Monitor marks the Ceph OSD Daemon **down** after the **mon_osd_report_timeout**, which is 900 seconds, elapses. A Ceph OSD Daemon sends a report to a Ceph Monitor when a reportable event such as a failure, a change in placement group stats, a change in **up_thru** or when it boots within 5 seconds.

You can change the Ceph OSD Daemon minimum report interval by setting the **osd_mon_report_interval** value at runtime:

**Syntax**

```
ceph config set osd osd_mon_report_interval TIME_IN_SECONDS
```

To get, set, and verify the config you can use the following example:

**Example**

```
[ceph: root@host01 /]# ceph config get osd osd_mon_report_interval
5
[ceph: root@host01 /]# ceph config set osd osd_mon_report_interval 20
[ceph: root@host01 /]# ceph config dump | grep osd
```

```
global                  advanced  osd_pool_default_crush_rule                  -1
osd                   basic     osd_memory_target                            4294967296
osd                   advanced  osd_mon_report_interval                      20
```

7.7. ADDITIONAL RESOURCES
See all the Red Hat Ceph Storage Ceph Monitor and OSD configuration options in Appendix G for specific option descriptions and usage.
CHAPTER 8. CEPH DEBUGGING AND LOGGING CONFIGURATION

As a storage administrator, you can increase the amount of debugging and logging information in cephadm to help diagnose problems with Red Hat Ceph Storage.

8.1. PREREQUISITES

- Red Hat Ceph Storage software is installed.

8.2. ADDITIONAL RESOURCES

- For more information about troubleshooting cephadm, see Cephadm troubleshooting in the Red Hat Ceph Storage Administration Guide.

- For more information about cephadm logging, see Cephadm operations in the Red Hat Ceph Storage Administration Guide.
APPENDIX A. GENERAL CONFIGURATION OPTIONS

These are the general configuration options for Ceph.

NOTE
Typically, these will be set automatically by deployment tools, such as cephadm.

fsid
Description
The file system ID. One per cluster.

Type
UUID
Required
No.
Default
N/A. Usually generated by deployment tools.

admin_socket
Description
The socket for executing administrative commands on a daemon, irrespective of whether Ceph monitors have established a quorum.

Type
String
Required
No
Default
/var/run/ceph/$cluster-$name.asok

pid_file
Description
The file in which the monitor or OSD will write its PID. For instance, /var/run/ceph/$cluster/$type.$id.pid will create /var/run/ceph/mon.a.pid for the mon with id a running in the ceph cluster. The pid file is removed when the daemon stops gracefully. If the process is not daemonized (meaning it runs with the -f or -d option), the pid file is not created.

Type
String
Required
No
Default
No

chdir
Description
The directory Ceph daemons change to once they are up and running. Default / directory recommended.

**Type**
- String

**Required**
- No

**Default**
- /

**max_open_files**

**Description**
If set, when the Red Hat Ceph Storage cluster starts, Ceph sets the `max_open_fds` at the OS level (that is, the max # of file descriptors). It helps prevent Ceph OSDs from running out of file descriptors.

**Type**
- 64-bit Integer

**Required**
- No

**Default**
- 0

**fatal_signal_handlers**

**Description**
If set, we will install signal handlers for SEGV, ABRT, BUS, ILL, FPE, XCPU, XFSZ, SYS signals to generate a useful log message.

**Type**
- Boolean

**Default**
- true
APPENDIX B. CEPH NETWORK CONFIGURATION OPTIONS

These are the common network configuration options for Ceph.

**public_network**  
Description  
The IP address and netmask of the public (front-side) network (for example, `192.168.0.0/24`). Set in `[global]`. You can specify comma-delimited subnets.  
Type  
`<ip-address>/<netmask> [, <ip-address>/<netmask>]`  
Required  
No  
Default  
N/A

**public_addr**  
Description  
The IP address for the public (front-side) network. Set for each daemon.  
Type  
IP Address  
Required  
No  
Default  
N/A

**cluster_network**  
Description  
The IP address and netmask of the cluster network (for example, `10.0.0.0/24`). Set in `[global]`. You can specify comma-delimited subnets.  
Type  
`<ip-address>/<netmask> [, <ip-address>/<netmask>]`  
Required  
No  
Default  
N/A

**cluster_addr**  
Description  
The IP address for the cluster network. Set for each daemon.  
Type  
Address  
Required  
No  
Default
ms_type

Description
The messenger type for the network transport layer. Red Hat supports the simple and the async messenger type using posix semantics.

Type
String.

Required
No.

Default
async+posix

ms_public_type

Description
The messenger type for the network transport layer of the public network. It operates identically to ms_type, but is applicable only to the public or front-side network. This setting enables Ceph to use a different messenger type for the public or front-side and cluster or back-side networks.

Type
String.

Required
No.

Default
None.

ms_cluster_type

Description
The messenger type for the network transport layer of the cluster network. It operates identically to ms_type, but is applicable only to the cluster or back-side network. This setting enables Ceph to use a different messenger type for the public or front-side and cluster or back-side networks.

Type
String.

Required
No.

Default
None.

Host options
You must declare at least one Ceph Monitor in the Ceph configuration file, with a mon addr setting under each declared monitor. Ceph expects a host setting under each declared monitor, metadata server and OSD in the Ceph configuration file.
IMPORTANT

Do not use localhost. Use the short name of the node, not the fully-qualified domain name (FQDN). Do not specify any value for host when using a third party deployment system that retrieves the node name for you.

mon_addr
Description
A list of <hostname>:<port> entries that clients can use to connect to a Ceph monitor. If not set, Ceph searches [mon.*] sections.

Type
String
Required
No
Default
N/A

host
Description
The host name. Use this setting for specific daemon instances (for example, [osd.0]).

Type
String
Required
Yes, for daemon instances.
Default
localhost

TCP options
Ceph disables TCP buffering by default.

ms_tcp_nodelay
Description
Ceph enables ms_tcp_nodelay so that each request is sent immediately (no buffering). Disabling Nagle’s algorithm increases network traffic, which can introduce congestion. If you experience large numbers of small packets, you may try disabling ms_tcp_nodelay, but be aware that disabling it will generally increase latency.

Type
Boolean
Required
No
Default
true

ms_tcp_rcvbuf
Description
The size of the socket buffer on the receiving end of a network connection. Disabled by default.

**Type**
- 32-bit Integer

**Required**
- No

**Default**
- 0

**ms_tcp_read_timeout**

**Description**
- If a client or daemon makes a request to another Ceph daemon and does not drop an unused connection, the *tcp read timeout* defines the connection as idle after the specified number of seconds.

**Type**
- Unsigned 64-bit Integer

**Required**
- No

**Default**
- 900 (15 minutes)

**Bind options**

The bind options configure the default port ranges for the Ceph OSD daemons. The default range is **6800:7100**. You can also enable Ceph daemons to bind to IPv6 addresses.

**ms_bind_port_min**

**Description**
- The minimum port number to which an OSD daemon will bind.

**Type**
- 32-bit Integer

**Default**
- 6800

**Required**
- No

**ms_bind_port_max**

**Description**
- The maximum port number to which an OSD daemon will bind.

**Type**
- 32-bit Integer

**Default**
7300  
Required  
No.

**ms_bind_ipv6**  
**Description**  
Enables Ceph daemons to bind to IPv6 addresses.  
**Type**  
Boolean  
**Default**  
false  
**Required**  
No

**Asynchronous messenger options**  
These Ceph messenger options configure the behavior of AsyncMessenger.

**ms_async_transport_type**  
**Description**  
Transport type used by the AsyncMessenger. Red Hat supports the posix setting, but does not support the dpdk or rdma settings at this time. POSIX uses standard TCP/IP networking and is the default value. Other transport types are experimental and are NOT supported.  
**Type**  
String  
**Required**  
No  
**Default**  
posix

**ms_async_op_threads**  
**Description**  
Initial number of worker threads used by each AsyncMessenger instance. This configuration setting SHOULD equal the number of replicas or erasure code chunks, but it may be set lower if the CPU core count is low or the number of OSDs on a single server is high.  
**Type**  
64-bit Unsigned Integer  
**Required**  
No  
**Default**  
3

**ms_async_max_op_threads**  
**Description**
The maximum number of worker threads used by each `AsyncMessenger` instance. Set to lower values if the OSD host has limited CPU count, and increase if Ceph is underutilizing CPUs are underutilized.

**Type**
- 64-bit Unsigned Integer

**Required**
- No

**Default**
- 5

**Description**
- Set to `true` to bind `AsyncMessenger` workers to particular CPU cores.

**Type**
- Boolean

**Required**
- No

**Default**
- `true`

**ms_async_affinity**

**Description**
- When `ms_async_set_affinity` is `true`, this string specifies how `AsyncMessenger` workers are bound to CPU cores. For example, `0,2` will bind workers #1 and #2 to CPU cores #0 and #2, respectively. **NOTE:** When manually setting affinity, make sure to not assign workers to virtual CPUs created as an effect of hyper threading or similar technology, because they are slower than physical CPU cores.

**Type**
- String

**Required**
- No

**Default**
- (empty)

**ms_async_send_inline**

**Description**
- Send messages directly from the thread that generated them instead of queuing and sending from the `AsyncMessenger` thread. This option is known to decrease performance on systems with a lot of CPU cores, so it's disabled by default.

**Type**
- Boolean

**Required**
- No

**Default**
- `false`
APPENDIX C. CEPH MONITOR CONFIGURATION OPTIONS

The following are Ceph monitor configuration options that can be set up during deployment.

**mon_initial_members**

*Description*

The IDs of initial monitors in a cluster during startup. If specified, Ceph requires an odd number of monitors to form an initial quorum (for example, 3).

*Type*

String

*Default*

None

**mon_force_quorum_join**

*Description*

Force monitor to join quorum even if it has been previously removed from the map

*Type*

Boolean

*Default*

False

**mon_dns_srv_name**

*Description*

The service name used for querying the DNS for the monitor hosts/addresses.

*Type*

String

*Default*

ceph-mon

**fsid**

*Description*

The cluster ID. One per cluster.

*Type*

UUID

*Required*

Yes.

*Default*

N/A. May be generated by a deployment tool if not specified.

**mon_data**

*Description*

The monitor’s data location.

*Type*

String
mon_data_size_warn
Description
Ceph issues a HEALTH_WARN status in the cluster log when the monitor’s data store reaches this threshold. The default value is 15GB.
Type
Integer
Default
15*1024*1024*1024*

mon_data_avail_warn
Description
Ceph issues a HEALTH_WARN status in the cluster log when the available disk space of the monitor’s data store is lower than or equal to this percentage.
Type
Integer
Default
30

mon_data_avail_crit
Description
Ceph issues a HEALTH_ERR status in the cluster log when the available disk space of the monitor’s data store is lower or equal to this percentage.
Type
Integer
Default
5

mon_warn_on_cache_pools_without_hit_sets
Description
Ceph issues a HEALTH_WARN status in the cluster log if a cache pool does not have the hit_set_type parameter set.
Type
Boolean
Default
True

mon_warn_on_crush_straw_calc_version_zero
Description
Ceph issues a HEALTH_WARN status in the cluster log if the CRUSH’s straw_calc_version is zero. See CRUSH tunables for details.
Type
**mon_warn_on_legacy_crush_tunables**

**Description**
Ceph issues a **HEALTH_WARN** status in the cluster log if CRUSH tunables are too old (older than **mon_min_crush_required_version**).

**Type**
Boolean

**Default**
True

**mon_crush_min_required_version**

**Description**
This setting defines the minimum tunable profile version required by the cluster.

**Type**
String

**Default**
firefly

**mon_warn_on_osd_down_out_interval_zero**

**Description**
Ceph issues a **HEALTH_WARN** status in the cluster log if the **mon_osd_down_out_interval** setting is zero, because the Leader behaves in a similar manner when the **noout** flag is set. Administrators find it easier to troubleshoot a cluster by setting the **noout** flag. Ceph issues the warning to ensure administrators know that the setting is zero.

**Type**
Boolean

**Default**
True

**mon_cache_target_full_warn_ratio**

**Description**
Ceph issues a warning when between the ratio of **cache_target_full** and **target_max_object**.

**Type**
Float

**Default**
0.66

**mon_health_data_update_interval**

**Description**
How often (in seconds) a monitor in the quorum shares its health status with its peers. A negative number disables health updates.
mon_health_to_clog
Description
This setting enables Ceph to send a health summary to the cluster log periodically.
Type
Boolean
Default
True

mon_health_detail_to_clog
Description
This setting enable Ceph to send a health details to the cluster log periodically.
Type
Boolean
Default
True

mon_op_complaint_time
Description
Number of seconds after which the Ceph Monitor operation is considered blocked after no updates.
Type
Integer
Default
30

mon_health_to_clog_tick_interval
Description
How often (in seconds) the monitor sends a health summary to the cluster log. A non-positive number disables it. If the current health summary is empty or identical to the last time, the monitor will not send the status to the cluster log.
Type
Integer
Default
3600

mon_health_to_clog_interval
Description
How often (in seconds) the monitor sends a health summary to the cluster log. A non-positive number disables it. The monitor will always send the summary to the cluster log.
mon_osd_full_ratio

Description
The percentage of disk space used before an OSD is considered full.

Type
Float
Default
.95

mon_osd_nearfull_ratio

Description
The percentage of disk space used before an OSD is considered nearfull.

Type
Float
Default
.85

mon_sync_trim_timeout

Description, Type
Double
Default
30.0

mon_sync_heartbeat_timeout

Description, Type
Double
Default
30.0

mon_sync_heartbeat_interval

Description, Type
Double
Default
5.0

mon_sync_backoff_timeout

Description, Type
Double
Default
mon_sync_timeout

Description
The number of seconds the monitor will wait for the next update message from its sync provider before it gives up and bootstraps again.

Type
Double

Default
30.0

mon_sync_max_retries

Description, Type
Integer

Default
5

mon_sync_max_payload_size

Description
The maximum size for a sync payload (in bytes).

Type
32-bit Integer

Default
1045676

paxos_max_join_drift

Description
The maximum Paxos iterations before we must first sync the monitor data stores. When a monitor finds that its peer is too far ahead of it, it will first sync with data stores before moving on.

Type
Integer

Default
10

paxos_stash_full_interval

Description
How often (in commits) to stash a full copy of the PaxosService state. Currently this setting only affects mds, mon, auth and mgr PaxosServices.

Type
Integer

Default
25

paxos_propose_interval
Gather updates for this time interval before proposing a map update.

Type
Double
Default
$1.0$

**paxos_min**
Description
The minimum number of paxos states to keep around
Type
Integer
Default
$500$

**paxos_min_wait**
Description
The minimum amount of time to gather updates after a period of inactivity.
Type
Double
Default
$0.05$

**paxos_trim_min**
Description
Number of extra proposals tolerated before trimming
Type
Integer
Default
$250$

**paxos_trim_max**
Description
The maximum number of extra proposals to trim at a time
Type
Integer
Default
$500$

**paxos_service_trim_min**
Description
The minimum amount of versions to trigger a trim ($0$ disables it)
Type
APPENDIX C. CEPH MONITOR CONFIGURATION OPTIONS

Integer
Default
250

**paxos_service_trim_max**

**Description**
The maximum amount of versions to trim during a single proposal (0 disables it)

**Type**
Integer

**Default**
500

**mon_max_log_epochs**

**Description**
The maximum amount of log epochs to trim during a single proposal

**Type**
Integer

**Default**
500

**mon_max_pgmap_epochs**

**Description**
The maximum amount of pgmap epochs to trim during a single proposal

**Type**
Integer

**Default**
500

**mon_mds_force_trim_to**

**Description**
Force monitor to trim mdsmaps to this point (0 disables it. dangerous, use with care)

**Type**
Integer

**Default**
0

**mon_osd_force_trim_to**

**Description**
Force monitor to trim osdmaps to this point, even if there is PGs not clean at the specified epoch (0 disables it. dangerous, use with care)

**Type**
Integer

**Default**
mon_osd_cache_size
Description
The size of osdmaps cache, not to rely on underlying store’s cache
Type
Integer
Default
10

mon_election_timeout
Description
On election proposer, maximum waiting time for all ACKs in seconds.
Type
Float
Default
5

mon_lease
Description
The length (in seconds) of the lease on the monitor’s versions.
Type
Float
Default
5

mon_lease_renew_interval_factor
Description
mon lease * mon lease renew interval factor will be the interval for the Leader to renew the other monitor’s leases. The factor should be less than 1.0.
Type
Float
Default
0.6

mon_lease_ack_timeout_factor
Description
The Leader will wait mon lease * mon lease ack timeout factor for the Providers to acknowledge the lease extension.
Type
Float
Default
2.0
mon_accept_timeout_factor
Description
The Leader will wait mon lease * mon accept timeout factor for the Requesters to accept a Paxos update. It is also used during the Paxos recovery phase for similar purposes.

Type
Float
Default
2.0

mon_min_osdmap_epochs
Description
Minimum number of OSD map epochs to keep at all times.

Type
32-bit Integer
Default
500

mon_max_pgmap_epochs
Description
Maximum number of PG map epochs the monitor should keep.

Type
32-bit Integer
Default
500

mon_max_log_epochs
Description
Maximum number of Log epochs the monitor should keep.

Type
32-bit Integer
Default
500

clock_offset
Description
How much to offset the system clock. See Clock.cc for details.

Type
Double
Default
0

mon_tick_interval
Description
A monitor’s tick interval in seconds.

**Type**
32-bit Integer

**Default**
5

**mon_clock_drift_allowed**

**Description**
The clock drift in seconds allowed between monitors.

**Type**
Float

**Default**
.050

**mon_clock_drift_warn_backoff**

**Description**
Exponential backoff for clock drift warnings.

**Type**
Float

**Default**
5

**mon_timecheck_interval**

**Description**
The time check interval (clock drift check) in seconds for the leader.

**Type**
Float

**Default**
300.0

**mon_timecheck_skew_interval**

**Description**
The time check interval (clock drift check) in seconds when in the presence of a skew in seconds for the Leader.

**Type**
Float

**Default**
30.0

**mon_max_osd**

**Description**
The maximum number of OSDs allowed in the cluster.

**Type**
mon_globalid_prealloc

Description
The number of global IDs to pre-allocate for clients and daemons in the cluster.

Type
32-bit Integer

Default
100

mon_sync_fs_threshold

Description
Synchronize with the filesystem when writing the specified number of objects. Set it to 0 to disable it.

Type
32-bit Integer

Default
5

mon_subscribe_interval

Description
The refresh interval, in seconds, for subscriptions. The subscription mechanism enables obtaining the cluster maps and log information.

Type
Double

Default
300

mon_stat_smooth_intervals

Description
Ceph will smooth statistics over the last N PG maps.

Type
Integer

Default
2

mon_probe_timeout

Description
Number of seconds the monitor will wait to find peers before bootstrapping.

Type
Double

Default
mon_daemon_bytes

Description
The message memory cap for metadata server and OSD messages (in bytes).

Type
64-bit Integer Unsigned

Default
400ul << 20

mon_max_log_entries_per_event

Description
The maximum number of log entries per event.

Type
Integer

Default
4096

mon_osd_prime_pg_temp

Description
Enables or disable priming the PGMap with the previous OSDs when an out OSD comes back into the cluster. With the true setting, the clients will continue to use the previous OSDs until the newly in OSDs as that PG peered.

Type
Boolean

Default
true

mon_osd_prime_pg_temp_max_time

Description
How much time in seconds the monitor should spend trying to prime the PGMap when an out OSD comes back into the cluster.

Type
Float

Default
0.5

mon_osd_prime_pg_temp_max_time_estimate

Description
Maximum estimate of time spent on each PG before we prime all PGs in parallel.

Type
Float

Default
0.25
mon_osd_allow_primary_affinity
Description
allow primary_affinity to be set in the osdmap.
Type
Boolean
Default
False

mon_osd_pool_ec_fast_read
Description
Whether turn on fast read on the pool or not. It will be used as the default setting of newly created erasure pools if fast_read is not specified at create time.
Type
Boolean
Default
False

mon_mds_skip_sanity
Description
Skip safety assertions on FSMap, in case of bugs where we want to continue anyway. Monitor terminates if the FSMap sanity check fails, but we can disable it by enabling this option.
Type
Boolean
Default
False

mon_max_mdsmap_epochs
Description
The maximum amount of mdsmap epochs to trim during a single proposal.
Type
Integer
Default
500

mon_config_key_max_entry_size
Description
The maximum size of config-key entry (in bytes).
Type
Integer
Default
4096

mon_scrub_interval
Description
How often, in seconds, the monitor scrub its store by comparing the stored checksums with the computed ones of all the stored keys.

**Type**
Integer

**Default**
3600*24

**mon_scrub_max_keys**

**Description**
The maximum number of keys to scrub each time.

**Type**
Integer

**Default**
100

**mon_compact_on_start**

**Description**
Compact the database used as Ceph Monitor store on ceph-mon start. A manual compaction helps to shrink the monitor database and improve its performance if the regular compaction fails to work.

**Type**
Boolean

**Default**
False

**mon_compact_on_bootstrap**

**Description**
Compact the database used as Ceph Monitor store on bootstrap. The monitor starts probing each other for creating a quorum after bootstrap. If it times out before joining the quorum, it will start over and bootstrap itself again.

**Type**
Boolean

**Default**
False

**mon_compact_on_trim**

**Description**
Compact a certain prefix (including paxos) when we trim its old states.

**Type**
Boolean

**Default**
True

**mon_cpu_threads**
Description
Number of threads for performing CPU intensive work on monitor.

Type
Boolean

Default
True

mon_osd_mapping_pgs_per_chunk
Description
We calculate the mapping from the placement group to OSDs in chunks. This option specifies the number of placement groups per chunk.

Type
Integer

Default
4096

mon_osd_max_split_count
Description
Largest number of PGs per "involved" OSD to let split create. When we increase the pg_num of a pool, the placement groups will be split on all OSDs serving that pool. We want to avoid extreme multipliers on PG splits.

Type
Integer

Default
300

rados_mon_op_timeout
Description
Number of seconds to wait for a response from the monitor before returning an error from a rados operation. 0 means at limit, or no wait time.

Type
Double

Default
0

Additional Resources
- Pool Values
- CRUSH tunables
APPENDIX D. CEPHX CONFIGURATION OPTIONS

The following are Cephx configuration options that can be set up during deployment.

auth_cluster_required

Description
If enabled, the Red Hat Ceph Storage cluster daemons, ceph-mon and ceph-osd, must authenticate with each other. Valid settings are cephx or none.

Type
String

Required
No

Default
cephx.

auth_service_required

Description
If enabled, the Red Hat Ceph Storage cluster daemons require Ceph clients to authenticate with the Red Hat Ceph Storage cluster in order to access Ceph services. Valid settings are cephx or none.

Type
String

Required
No

Default
cephx.

auth_client_required

Description
If enabled, the Ceph client requires the Red Hat Ceph Storage cluster to authenticate with the Ceph client. Valid settings are cephx or none.

Type
String

Required
No

Default
cephx.

keyring

Description
The path to the keyring file.

Type
String

Required
APPENDIX D. CEPHX CONFIGURATION OPTIONS

No
Default

/etc/ceph/$cluster.$name.keyring,/etc/ceph/$cluster.keyring,/etc/ceph/keyring,/etc/ceph/keyring.bin

keyfile
Description
The path to a key file (that is, a file containing only the key).
Type
String
Required
No
Default
None

key
Description
The key (that is, the text string of the key itself). Not recommended.
Type
String
Required
No
Default
None

ceph-mon
Location
$mon_data/keyring
Capabilities
mon 'allow *'

ceph-osd
Location
$osd_data/keyring
Capabilities
mon 'allow profile osd' osd 'allow *'

radosgw
Location
$rgw_data/keyring
Capabilities
mon 'allow rwx' osd 'allow rwx'

cephx_require_signatures
Description
If set to true, Ceph requires signatures on all message traffic between the Ceph client and the Red Hat Ceph Storage cluster, and between daemons comprising the Red Hat Ceph Storage cluster.

Type
Boolean
Required
No
Default
false

cephx_cluster_require_signatures
Description
If set to true, Ceph requires signatures on all message traffic between Ceph daemons comprising the Red Hat Ceph Storage cluster.

Type
Boolean
Required
No
Default
false

cephx_service_require_signatures
Description
If set to true, Ceph requires signatures on all message traffic between Ceph clients and the Red Hat Ceph Storage cluster.

Type
Boolean
Required
No
Default
false

cephx_sign_messages
Description
If the Ceph version supports message signing, Ceph will sign all messages so they cannot be spoofed.

Type
Boolean
Default
true

auth_service_ticket_ttl
Description
When the Red Hat Ceph Storage cluster sends a Ceph client a ticket for authentication, the cluster assigns the ticket a time to live.

**Type**
- Double

**Default**
- 60*60
APPENDIX E. POOLS, PLACEMENT GROUPS, AND CRUSH CONFIGURATION OPTIONS

The Ceph options that govern pools, placement groups, and the CRUSH algorithm.

`mon_allow_pool_delete`

**Description**
Allows a monitor to delete a pool. In RHCS 3 and later releases, the monitor cannot delete the pool by default as an added measure to protect data.

**Type**
Boolean

**Default**
false

`mon_max_pool_pg_num`

**Description**
The maximum number of placement groups per pool.

**Type**
Integer

**Default**
65536

`mon_pg_create_interval`

**Description**
Number of seconds between PG creation in the same Ceph OSD Daemon.

**Type**
Float

**Default**
30.0

`mon_pg_stuck_threshold`

**Description**
Number of seconds after which PGs can be considered as being stuck.

**Type**
32-bit Integer

**Default**
300

`mon_pg_min_inactive`

**Description**
Ceph issues a `HEALTH_ERR` status in the cluster log if the number of PGs that remain inactive longer than the `mon_pg_stuck_threshold` exceeds this setting. The default setting is one PG. A non-positive number disables this setting.

**Type**
mon_pg_warn_min_per_osd
Description
Ceph issues a **HEALTH_WARN** status in the cluster log if the average number of PGs per OSD in the cluster is less than this setting. A non-positive number disables this setting.
Type
Integer
Default
30

mon_pg_warn_max_per_osd
Description
Ceph issues a **HEALTH_WARN** status in the cluster log if the average number of PGs per OSD in the cluster is greater than this setting. A non-positive number disables this setting.
Type
Integer
Default
300

mon_pg_warn_min_objects
Description
Do not warn if the total number of objects in the cluster is below this number.
Type
Integer
Default
1000

mon_pg_warn_min_pool_objects
Description
Do not warn on pools whose object number is below this number.
Type
Integer
Default
1000

mon_pg_check_down_all_threshold
Description
The threshold of **down** OSDs by percentage after which Ceph checks all PGs to ensure they are not stuck or stale.
Type
Float
mon_pg_warn_max_object_skew

Description
Ceph issue a HEALTH_WARN status in the cluster log if the average number of objects in a pool is greater than mon_pg_warn_max_object_skew times the average number of objects for all pools. A non-positive number disables this setting.

Type
Float

Default
10

mon_delta_reset_interval

Description
The number of seconds of inactivity before Ceph resets the PG delta to zero. Ceph keeps track of the delta of the used space for each pool to aid administrators in evaluating the progress of recovery and performance.

Type
Integer

Default
10

mon_osd_max_op_age

Description
The maximum age in seconds for an operation to complete before issuing a HEALTH_WARN status.

Type
Float

Default
32.0

osd.pg_bits

Description
Placement group bits per Ceph OSD Daemon.

Type
32-bit Integer

Default
6

osd.pgp_bits

Description
The number of bits per Ceph OSD Daemon for Placement Groups for Placement purpose (PGPs).

Type
32-bit Integer
Default
6

**osd_crush_chooseleaf_type**

**Description**
The bucket type to use for `chooseleaf` in a CRUSH rule. Uses ordinal rank rather than name.

**Type**
32-bit Integer

**Default**
1. Typically a host containing one or more Ceph OSD Daemons.

**osd_pool_default_crush_replicated_ruleset**

**Description**
The default CRUSH ruleset to use when creating a replicated pool.

**Type**
8-bit Integer

**Default**
0

**osd_pool_erosure_code_stripe_unit**

**Description**
Sets the default size, in bytes, of a chunk of an object stripe for erasure coded pools. Every object of size S will be stored as N stripes, with each data chunk receiving `stripe unit` bytes. Each stripe of \( N \times \text{stripe unit} \) bytes will be encoded/decoded individually. This option can be overridden by the `stripe_unit` setting in an erasure code profile.

**Type**
Unsigned 32-bit Integer

**Default**
4096

**osd_pool_default_size**

**Description**
Sets the number of replicas for objects in the pool. The default value is the same as `ceph osd pool set {pool-name} size {size}`.

**Type**
32-bit Integer

**Default**
3

**osd_pool_default_min_size**

**Description**
Sets the minimum number of written replicas for objects in the pool in order to acknowledge a write operation to the client. If the minimum is not met, Ceph will not acknowledge the write to the client. This setting ensures a minimum number of replicas when operating in `degraded` mode.
Type
32-bit Integer

Default
0, which means no particular minimum. If 0, minimum is size - (size / 2).

**osd_pool_default_pg_num**

Description
The default number of placement groups for a pool. The default value is the same as **pg_num** with **mkpool**.

Type
32-bit Integer

Default
8

**osd_pool_default_pgp_num**

Description
The default number of placement groups for placement for a pool. The default value is the same as **pgp_num** with **mkpool**. PG and PGP should be equal.

Type
32-bit Integer

Default
8

**osd_pool_default_flags**

Description
The default flags for new pools.

Type
32-bit Integer

Default
0

**osd_max_pglsls**

Description
The maximum number of placement groups to list. A client requesting a large number can tie up the Ceph OSD Daemon.

Type
Unsigned 64-bit Integer

Default
1024

Note
Default should be fine.

**osd_min_pg_log_entries**

Description
The minimum number of placement group logs to maintain when trimming log files.

**Type**
32-bit Int Unsigned

**Default**
1000

**osd_default_data_pool_replay_window**

**Description**
The time, in seconds, for an OSD to wait for a client to replay a request.

**Type**
32-bit Integer

**Default**
45
APPENDIX F. OBJECT STORAGE DAEMON (OSD) CONFIGURATION OPTIONS

The following are Ceph Object Storage Daemon (OSD) configuration options that can be set during deployment.

**osd_uuid**

**Description**
The universally unique identifier (UUID) for the Ceph OSD.

**Type**
UUID

**Default**
The UUID.

**Note**
The `osd uuid` applies to a single Ceph OSD. The `fsid` applies to the entire cluster.

**osd_data**

**Description**
The path to the OSD’s data. You must create the directory when deploying Ceph. Mount a drive for OSD data at this mount point.

**IMPORTANT:** Red Hat does not recommend changing the default.

**Type**
String

**Default**
`/var/lib/ceph/osd/$cluster-$id`

**osd_max_write_size**

**Description**
The maximum size of a write in megabytes.

**Type**
32-bit Integer

**Default**
90

**osd_client_message_size_cap**

**Description**
The largest client data message allowed in memory.

**Type**
64-bit Integer Unsigned

**Default**
500MB default. **500*1024L*1024L**

**osd_class_dir**
Description
The class path for RADOS class plug-ins.

Type
String

Default
$libdir/rados-classes

osd_max_scrubs

Description
The maximum number of simultaneous scrub operations for a Ceph OSD.

Type
32-bit Int

Default
1

osd_scrub_thread_timeout

Description
The maximum time in seconds before timing out a scrub thread.

Type
32-bit Integer

Default
60

osd_scrub_finalize_thread_timeout

Description
The maximum time in seconds before timing out a scrub finalize thread.

Type
32-bit Integer

Default
60*10

osd_scrub_begin_hour

Description
The earliest hour that light or deep scrubbing can begin. It is used with the osd scrub end hour parameter to define a scrubbing time window and allows constraining scrubbing to off-peak hours. The setting takes an integer to specify the hour on the 24-hour cycle where 0 represents the hour from 12:01 a.m. to 1:00 a.m., 13 represents the hour from 1:01 p.m. to 2:00 p.m., and so on.

Type
32-bit Integer

Default
0 for 12:01 to 1:00 a.m.

osd_scrub_end_hour

Description
The latest hour that light or deep scrubbing can begin. It is used with the `osd scrub begin hour` parameter to define a scrubbing time window and allows constraining scrubbing to off-peak hours. The setting takes an integer to specify the hour on the 24-hour cycle where 0 represents the hour from 12:01 a.m. to 1:00 a.m., 13 represents the hour from 1:01 p.m. to 2:00 p.m., and so on. The `end` hour must be greater than the `begin` hour.

**Type**

32-bit Integer

**Default**

24 for 11:01 p.m. to 12:00 a.m.

---

**osd_scrub_load_threshold**

**Description**

The maximum load. Ceph will not scrub when the system load (as defined by the `getloadavg()` function) is higher than this number. Default is 0.5.

**Type**

Float

**Default**

0.5

---

**osd_scrub_min_interval**

**Description**

The minimum interval in seconds for scrubbing the Ceph OSD when the Red Hat Ceph Storage cluster load is low.

**Type**

Float

**Default**

Once per day. 60*60*24

---

**osd_scrub_max_interval**

**Description**

The maximum interval in seconds for scrubbing the Ceph OSD irrespective of cluster load.

**Type**

Float

**Default**

Once per week. 7*60*60*24

---

**osd_scrub_interval_randomize_ratio**

**Description**

Takes the ratio and randomizes the scheduled scrub between `osd scrub min interval` and `osd scrub max interval`.

**Type**

Float

**Default**

0.5
mon_warn_not_scrubbed

Description

Number of seconds after `osd_scrub_interval` to warn about any PGs that were not scrubbed.

Type

Integer

Default

0 (no warning).

osd_scrub_chunk_min

Description

The object store is partitioned into chunks which end on hash boundaries. For chunky scrubs, Ceph scrubs objects one chunk at a time with writes blocked for that chunk. The `osd_scrub_chunk_min` setting represents the minimum number of chunks to scrub.

Type

32-bit Integer

Default

5

osd_scrub_chunk_max

Description

The maximum number of chunks to scrub.

Type

32-bit Integer

Default

25

osd_scrub_sleep

Description

The time to sleep between deep scrub operations.

Type

Float

Default

0 (or off).

osd_scrub_during_recovery

Description

Allows scrubbing during recovery.

Type

Bool

Default

false

osd_scrub_invalid_stats
Forces extra scrub to fix stats marked as invalid.

**Type**
- **Bool**

**Default**
- **true**

**osd_scrub_priority**

**Description**
- Controls queue priority of scrub operations versus client I/O.

**Type**
- Unsigned 32-bit Integer

**Default**
- **5**

**osd_scrub_cost**

**Description**
- Cost of scrub operations in megabytes for queue scheduling purposes.

**Type**
- Unsigned 32-bit Integer

**Default**
- **50 << 20**

**osd_deep_scrub_interval**

**Description**
- The interval for deep scrubbing, that is fully reading all data. The `osd scrub load threshold` parameter does not affect this setting.

**Type**
- Float

**Default**
- Once per week. **60*60*24*7**

**osd_deep_scrub_stride**

**Description**
- Read size when doing a deep scrub.

**Type**
- 32-bit Integer

**Default**
- **512 KB. 524288**

**mon_warn_not_deep_scrubbed**

**Description**
- Number of seconds after `osd_deep_scrub_interval` to warn about any PGs that were not scrubbed.

**Type**
Integer
Default
0 (no warning).

**osd_deep_scrub_randomize_ratio**

**Description**
The rate at which scrubs will randomly become deep scrubs (even before **osd_deep_scrub_interval** has passed).

**Type**
Float

**Default**
0.15 or 15%.

**osd_deep_scrub_update_digest_min_age**

**Description**
How many seconds old objects must be before scrub updates the whole-object digest.

**Type**
Integer

**Default**
120 (2 hours).

**osd_op_num_shards**

**Description**
The number of shards for client operations.

**Type**
32-bit Integer

**Default**
0

**osd_op_num_threads_per_shard**

**Description**
The number of threads per shard for client operations.

**Type**
32-bit Integer

**Default**
0

**osd_op_num_shards_hdd**

**Description**
The number of shards for HDD operations.

**Type**
32-bit Integer

**Default**
**osd_op_num_threads_per_shard_hdd**

**Description**
The number of threads per shard for HDD operations.

**Type**
32-bit Integer

**Default**
1

**osd_op_num_shards_ssd**

**Description**
The number of shards for SSD operations.

**Type**
32-bit Integer

**Default**
8

**osd_op_num_threads_per_shard_ssd**

**Description**
The number of threads per shard for SSD operations.

**Type**
32-bit Integer

**Default**
2

**osd_client_op_priority**

**Description**
The priority set for client operations. It is relative to **osd recovery op priority**.

**Type**
32-bit Integer

**Default**
63

**Valid Range**
1-63

**osd_recovery_op_priority**

**Description**
The priority set for recovery operations. It is relative to **osd client op priority**.

**Type**
32-bit Integer

**Default**
3
Valid Range
1-63

**osd_op_thread_timeout**

Description
The Ceph OSD operation thread timeout in seconds.

Type
32-bit Integer

Default
30

**osd_op_complaint_time**

Description
An operation becomes complaint worthy after the specified number of seconds have elapsed.

Type
Float

Default
30

**osd_disk_threads**

Description
The number of disk threads, which are used to perform background disk intensive OSD operations such as scrubbing and snap trimming.

Type
32-bit Integer

Default
1

**osd_disk_thread_ioprio_class**

Description
Sets the `ioprio_set(2)` I/O scheduling class for the disk thread. Acceptable values are:

- **idle**
- **be**
- **rt** + The idle class means the disk thread will have lower priority than any other thread in the OSD. This is useful to slow down scrubbing on an OSD that is busy handling client operations. + The be class is the default and is the same priority as all other threads in the OSD. + The rt class means the disk thread will have precedence over all other threads in the OSD. This is useful if scrubbing is much needed and must make progress at the expense of client operations.

Type
String

Default
osd_disk_thread_ioprio_priority

Description
It sets the `ioprio_set(2)` I/O scheduling priority of the disk thread ranging from 0 (highest) to 7 (lowest). If all OSDs on a given host were in class `idle` and compete for I/O due to controller congestion, it can be used to lower the disk thread priority of one OSD to 7 so that another OSD with priority 0 can potentially scrub faster.

Type
Integer in the range of 0 to 7 or -1 if not to be used.

Default
-1

IMPORTANT
The osd disk thread ioprio class and osd disk thread ioprio priority options will only be used if both are set to a non default value. In addition, it only works with the Linux Kernel CFQ scheduler.

osd_op_history_size

Description
The maximum number of completed operations to track.

Type
32-bit Unsigned Integer

Default
20

osd_op_history_duration

Description
The oldest completed operation to track.

Type
32-bit Unsigned Integer

Default
600

osd_op_log_threshold

Description
How many operations logs to display at once.

Type
32-bit Integer

Default
5

osd_op_timeout
**osd_op_timeout**

Description:
The time in seconds after which running OSD operations time out.

Type:
Integer

Default:
0

**IMPORTANT**

Do not set the **osd_op_timeout** option unless your clients can handle the consequences. For example, setting this parameter on clients running in virtual machines can lead to data corruption because the virtual machines interpret this timeout as a hardware failure.

**osd_max_backfills**

Description:
The maximum number of backfill operations allowed to or from a single OSD.

Type:
64-bit Unsigned Integer

Default:
1

**osd_backfill_scan_min**

Description:
The minimum number of objects per backfill scan.

Type:
32-bit Integer

Default:
64

**osd_backfill_scan_max**

Description:
The maximum number of objects per backfill scan.

Type:
32-bit Integer

Default:
512

**osd_backfill_full_ratio**

Description:
Refuse to accept backfill requests when the Ceph OSD's full ratio is above this value.

Type:
Float

Default:
osd_backfill_retry_interval

Description
The number of seconds to wait before retrying backfill requests.

Type
Double

Default
10.0

osd_map_dedup

Description
Enable removing duplicates in the OSD map.

Type
Boolean

Default
true

osd_map_cache_size

Description
The size of the OSD map cache in megabytes.

Type
32-bit Integer

Default
50

osd_map_cache_bl_size

Description
The size of the in-memory OSD map cache in OSD daemons.

Type
32-bit Integer

Default
50

osd_map_cache_bl_inc_size

Description
The size of the in-memory OSD map cache incrementals in OSD daemons.

Type
32-bit Integer

Default
100

osd_map_message_max
Description
   The maximum map entries allowed per MOSDMap message.
Type
   32-bit Integer
Default
   40

```
        osd_snap_trim_thread_timeout
        Description
            The maximum time in seconds before timing out a snap trim thread.
Type
   32-bit Integer
Default
    60*60*1
```

```
        osd_pg_max_concurrent_snap_trims
        Description
            The max number of parallel snap trims/PG. This controls how many objects per PG to trim at once.
Type
   32-bit Integer
Default
    2
```

```
        osd_snap_trim_sleep
        Description
            Insert a sleep between every trim operation a PG issues.
Type
   32-bit Integer
Default
    0
```

```
        osd_max_trimming_pgs
        Description
            The max number of trimming PGs
Type
   32-bit Integer
Default
    2
```

```
        osd_backlog_thread_timeout
        Description
            The maximum time in seconds before timing out a backlog thread.
```
Type 32-bit Integer
Default 60*60*1

osd_default_notify_timeout
Description
The OSD default notification timeout (in seconds).
Type 32-bit Integer Unsigned
Default 30

osd_check_for_log_corruption
Description
Check log files for corruption. Can be computationally expensive.
Type Boolean
Default false

osd_remove_thread_timeout
Description
The maximum time in seconds before timing out a remove OSD thread.
Type 32-bit Integer
Default 60*60

osd_command_thread_timeout
Description
The maximum time in seconds before timing out a command thread.
Type 32-bit Integer
Default 10*60

osd_command_max_records
Description
Limits the number of lost objects to return.
Type 32-bit Integer
Default
osd_auto_upgrade_tmap

Description
Uses tmap for omap on old objects.

Type
Boolean
Default
true

osd_tmapput_sets_users_tmap

Description
Uses tmap for debugging only.

Type
Boolean
Default
false

osd_preserve_trimmed_log

Description
Preserves trimmed log files, but uses more disk space.

Type
Boolean
Default
false

osd_recovery_delay_start

Description
After peering completes, Ceph delays for the specified number of seconds before starting to recover objects.

Type
Float
Default
0

osd_recovery_max_active

Description
The number of active recovery requests per OSD at one time. More requests will accelerate recovery, but the requests place an increased load on the cluster.

Type
32-bit Integer
Default
3
**osd_recovery_max_chunk**

**Description**
The maximum size of a recovered chunk of data to push.

**Type**
64-bit Integer Unsigned

**Default**
$8 \ll 20$

**osd_recovery_threads**

**Description**
The number of threads for recovering data.

**Type**
32-bit Integer

**Default**
1

**osd_recovery_thread_timeout**

**Description**
The maximum time in seconds before timing out a recovery thread.

**Type**
32-bit Integer

**Default**
30

**osd_recover_clone_overlap**

**Description**
Preserves clone overlap during recovery. Should always be set to **true**.

**Type**
Boolean

**Default**
true

**rados_osd_op_timeout**

**Description**
Number of seconds that RADOS waits for a response from the OSD before returning an error from a RADOS operation. A value of 0 means no limit.

**Type**
Double

**Default**
0
APPENDIX G. CEPH MONITOR AND OSD CONFIGURATION OPTIONS

When modifying heartbeat settings, include them in the [global] section of the Ceph configuration file.

mon_osd_min_up_ratio
  Description
    The minimum ratio of up Ceph OSD Daemons before Ceph will mark Ceph OSD Daemons down.
  Type
    Double
  Default
    .3

mon_osd_min_in_ratio
  Description
    The minimum ratio of in Ceph OSD Daemons before Ceph will mark Ceph OSD Daemons out.
  Type
    Double
  Default
    .3

mon_osd_laggy_halflife
  Description
    The number of seconds laggy estimates will decay.
  Type
    Integer
  Default
    60*60

mon_osd_laggy_weight
  Description
    The weight for new samples in laggy estimation decay.
  Type
    Double
  Default
    0.3

mon_osd_laggy_max_interval
  Description
    Maximum value of laggy_interval in laggy estimations (in seconds). The monitor uses an adaptive approach to evaluate the laggy_interval of a certain OSD. This value will be used to calculate the grace time for that OSD.
  Type
    Integer
mon_osd_adjust_heartbeat_grace
Description
If set to true, Ceph will scale based on laggy estimations.
Type
Boolean
Default
true

mon_osd_adjust_down_out_interval
Description
If set to true, Ceph will scale based on laggy estimations.
Type
Boolean
Default
true

mon_osd_auto_mark_in
Description
Ceph will mark any booting Ceph OSD Daemons as in the Ceph Storage Cluster.
Type
Boolean
Default
false

mon_osd_auto_mark_auto_out_in
Description
Ceph will mark booting Ceph OSD Daemons auto marked out of the Ceph Storage Cluster as in the cluster.
Type
Boolean
Default
true

mon_osd_auto_mark_new_in
Description
Ceph will mark booting new Ceph OSD Daemons as in the Ceph Storage Cluster.
Type
Boolean
Default
true
mon_osd_down_out_interval
Description
The number of seconds Ceph waits before marking a Ceph OSD Daemon down and out if it does not respond.
Type
32-bit Integer
Default
600

mon_osd_downout_subtree_limit
Description
The largest CRUSH unit type that Ceph will automatically mark out.
Type
String
Default
rack

mon_osd_reporter_subtree_level
Description
This setting defines the parent CRUSH unit type for the reporting OSDs. The OSDs send failure reports to the monitor if they find an unresponsive peer. The monitor may mark the reported OSD down and then out after a grace period.
Type
String
Default
host

mon_osd_report_timeout
Description
The grace period in seconds before declaring unresponsive Ceph OSD Daemons down.
Type
32-bit Integer
Default
900

mon_osd_min_down_reporters
Description
The minimum number of Ceph OSD Daemons required to report a down Ceph OSD Daemon.
Type
32-bit Integer
Default
2

osd_heartbeat_address
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>osd_heartbeat_interval</td>
<td>How often a Ceph OSD Daemon pings its peers (in seconds).</td>
<td>32-bit Integer</td>
<td>6</td>
</tr>
<tr>
<td>osd_heartbeat_grace</td>
<td>The elapsed time when a Ceph OSD Daemon has not shown a heartbeat that the Ceph Storage Cluster considers it <strong>down</strong>.</td>
<td>32-bit Integer</td>
<td>20</td>
</tr>
<tr>
<td>osd_mon_heartbeat_interval</td>
<td>How often the Ceph OSD Daemon pings a Ceph Monitor if it has no Ceph OSD Daemon peers.</td>
<td>32-bit Integer</td>
<td>30</td>
</tr>
<tr>
<td>osd_mon_report_interval_max</td>
<td>The maximum time in seconds that a Ceph OSD Daemon can wait before it must report to a Ceph Monitor.</td>
<td>32-bit Integer</td>
<td>120</td>
</tr>
<tr>
<td>osd_mon_report_interval_min</td>
<td>The minimum number of seconds a Ceph OSD Daemon may wait from startup or another Red Hat Ceph Storage 5 Configuration Guide</td>
<td>32-bit Integer</td>
<td>120</td>
</tr>
</tbody>
</table>
The minimum number of seconds a Ceph OSD Daemon may wait from startup or another reportable event before reporting to a Ceph Monitor.

**Type**
32-bit Integer

**Default**
5

**Valid Range**
Should be less than `osd mon report interval max`

**osd_mon_ack_timeout**

**Description**
The number of seconds to wait for a Ceph Monitor to acknowledge a request for statistics.

**Type**
32-bit Integer

**Default**
30
APPENDIX H. CEPH SCRUBBING OPTIONS

Ceph ensures data integrity by scrubbing placement groups. The following are the Ceph scrubbing options that you can adjust to increase or decrease scrubbing operations.

**osd_max_scrubs**

**Description**
The maximum number of simultaneous scrub operations for a Ceph OSD Daemon.

**Type**
integer

**Default**
1

**osd_scrub_begin_hour**

**Description**
The specific hour at which the scrubbing begins. Along with **osd_scrub_end_hour**, you can define a time window in which the scrubs can happen. Use **osd_scrub_begin_hour = 0** and **osd_scrub_end_hour = 0** to allow scrubbing the entire day.

**Type**
integer

**Default**
0

**Allowed range**
[0, 23]

**osd_scrub_end_hour**

**Description**
The specific hour at which the scrubbing ends. Along with **osd_scrub_begin_hour**, you can define a time window, in which the scrubs can happen. Use **osd_scrub_begin_hour = 0** and **osd_scrub_end_hour = 0** to allow scrubbing for the entire day.

**Type**
integer

**Default**
0

**Allowed range**
[0, 23]

**osd_scrub_begin_week_day**

**Description**
The specific day on which the scrubbing begins. 0 = Sunday, 1 = Monday, etc. Along with “**osd_scrub_end_week_day**”, you can define a time window in which scrubs can happen. Use **osd_scrub_begin_week_day = 0** and **osd_scrub_end_week_day = 0** to allow scrubbing for the entire week.

**Type**
integer
osd_scrub_end_week_day

Description
This defines the day on which the scrubbing ends. 0 = Sunday, 1 = Monday, etc. Along with osd_scrub_begin_week_day, they define a time window, in which the scrubs can happen. Use osd_scrub_begin_week_day = 0 and osd_scrub_end_week_day = 0 to allow scrubbing for the entire week.

Type
integer

Default
0

Allowed range
[0, 6]

osd_scrub_during_recovery

Description
Allow scrub during recovery. Setting this to false disables scheduling new scrub, and deep-scrub, while there is an active recovery. The already running scrubs continue which is useful to reduce load on busy storage clusters.

Type
boolean

Default
false

osd_scrub_load_threshold

Description
The normalized maximum load. Scrubbing does not happen when the system load, as defined by getloadavg() / number of online CPUs, is higher than this defined number.

Type
float

Default
0.5

osd_scrub_min_interval

Description
The minimal interval in seconds for scrubbing the Ceph OSD daemon when the Ceph storage Cluster load is low.

Type
float

Default
1 day
osd_scrub_max_interval

Description
The maximum interval in seconds for scrubbing the Ceph OSD daemon irrespective of cluster load.

Type
float

Default
7 days

osd_scrub_chunk_min

Description
The minimal number of object store chunks to scrub during a single operation. Ceph blocks writes to a single chunk during scrub.

type
integer

Default
5

osd_scrub_chunk_max

Description
The maximum number of object store chunks to scrub during a single operation.

type
integer

Default
25

osd_scrub_sleep

Description
Time to sleep before scrubbing the next group of chunks. Increasing this value slows down the overall rate of scrubbing, so that client operations are less impacted.

type
float

Default
0.0

osd_deep_scrub_interval

Description
The interval for deep scrubbing, fully reading all data. The osd_scrub_load_threshold does not affect this setting.

type
float

Default
7 days
**osd_scrub_interval_randomize_ratio**

*Description*
Add a random delay to `osd_scrub_min_interval` when scheduling the next scrub job for a placement group. The delay is a random value less than `osd_scrub_min_interval * osd_scrub_interval_randomized_ratio`. The default setting spreads scrubs throughout the allowed time window of `[1, 1.5] * osd_scrub_min_interval`.

*type*
float

*Default*
0.5

**osd_deep_scrub_stride**

*Description*
Read size when doing a deep scrub.

*type*
size

*Default*
512 KB

**osd_scrub_auto_repair_num_errors**

*Description*
Auto repair does not occur if more than this many errors are found.

*type*
integer

*Default*
5

**osd_scrub_auto_repair**

*Description*
Setting this to `true` enables automatic Placement Group (PG) repair when errors are found by scrubs or deep-scrubs. However, if more than `osd_scrub_auto_repair_num_errors` errors are found, a repair is NOT performed.

*type*
boolean

*Default*
false
APPENDIX I. BLUESTORE CONFIGURATION OPTIONS

The following are Ceph BlueStore configuration options that can be configured during deployment.

NOTE
This list is not complete.

rocksdb_cache_size
Description
The size of the RocksDB cache in MB.
Type
32-bit Integer
Default
512