# Red Hat Ceph Storage 1.3
# Configuration Guide

Configuration settings for Red Hat Ceph Storage

Red Hat Ceph Storage Documentation Team

# Red Hat Ceph Storage 1.3 Configuration Guide

Configuration settings for Red Hat Ceph Storage

## Legal Notice

## Abstract

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

# Table of Contents

# CHAPTER 1. CONFIGURATION REFERENCE

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 data (including journals)
- Other runtime options

A deployment tool such as **`ceph-deploy`** 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 (that is, 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 host.

## 1.1. GENERAL RECOMMENDATIONS

You may maintain a Ceph configuration file anywhere you like, but Red Hat recommends having an administration node where you maintain a master copy of the Ceph configuration file.

When you make changes to the Ceph configuration file, it is a good practice to push the updated configuration file to your Ceph nodes to maintain consistency:

```
ceph-deploy --overwrite-conf config push {node1}[, {node2}, ...]
```

If the Ceph nodes have a more recent copy, pull the Ceph configuration file to your administration node:

```
ceph-deploy --overwrite-conf config pull {node}
```

The commands in this reference assume you have an administration keyring. You will need to change the permissions to enable the current user or use **sudo** with the **ceph** command to access **`ceph.client.admin.keyring`**.

## 1.2. CONFIGURATION FILE STRUCTURE

The Ceph configuration file configures Ceph daemons at start time—overriding default values. Ceph configuration files use an *ini* style syntax. You can add comments by preceding comments with a pound sign (#) or a semi-colon (;). For example:

```
# <--A number (#) sign precedes a comment.
```

```
; A comment may be anything.
# Comments always follow a semi-colon (;) or a pound (#) on each line.
# The end of the line terminates a comment.
# We recommend that you provide comments in your configuration file(s).
```

The configuration file can configure all Ceph daemons in a Ceph storage cluster or all Ceph daemons of a particular type at start time. To configure a series of daemons, the settings must be included under the processes that will receive the configuration as follows:

**[global]**

**Description**

Settings under **`[global]`** affect all daemons in a Ceph Storage Cluster.

**Example**

**`auth supported = cephx`**

**[osd]**

**Description**

Settings under **`[osd]`** affect all **`ceph-osd`** daemons in the Ceph storage cluster, and override the same setting in **`[global]`**.

**Example**

**`osd journal size = 1000`**

**[mon]**

**Description**

Settings under **`[mon]`** affect all **`ceph-mon`** daemons in the Ceph storage cluster, and override the same setting in **`[global]`**.

**Example**

**`mon host = hostname1,hostname2,hostname3mon addr = 10.0.0.101:6789`**

**[client]**

**Description**

Settings under **`[client]`** affect all Ceph clients (for example, mounted Ceph block devices, Ceph object gateways, and so on).

**Example**

**`log file = /var/log/ceph/radosgw.log`**

Global settings affect all instances of all daemon in the Ceph storage cluster. Use the **`[global]`** setting for values that are common for all daemons in the Ceph storage cluster. You can override each **`[global]`** setting by:

1. Changing the setting in a particular process type (for example, **[osd]**, **[mon]**).

2. Changing the setting in a particular process (for example, **[osd.1]** ).

Overriding a global setting affects all child processes, except those that you specifically override in a particular daemon.

A typical global setting involves activating authentication. For example:

```
[global]
#Enable authentication between hosts within the cluster.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

You can specify settings that apply to a particular type of daemon. When you specify settings under **[osd]** or **[mon]** without specifying a particular instance, the setting will apply to all OSD or monitor daemons respectively.

A typical daemon-wide setting involves setting journal sizes, filestore settings, and so on For example:

```
[osd]
osd_journal_size = 1000
```

You can specify settings for particular instances of a daemon. You may specify an instance by entering its type, delimited by a period (.) and by the instance ID. The instance ID for a Ceph OSD daemons is always numeric, but it may be alphanumeric for Ceph monitors.

```
[osd.1]
# settings affect osd.1 only.

[mon.a]
# settings affect mon.a only.
```

The default Ceph configuration file locations in sequential order include:

1. **$CEPH_CONF** (the path following the **$CEPH_CONF** environment variable)

2. **-c path/path** (the **-c** command line argument)

3. **/etc/ceph/ceph.conf**

4. **~/.ceph/config**

5. **./ceph.conf** (in the current working directory)

A typical Ceph configuration file has at least the following settings:

```
[global]
fsid = {cluster-id}
mon_initial_members = {hostname}[, {hostname}]
mon_host = {ip-address}[, {ip-address}]

#All clusters have a front-side public network.
#If you have two NICs, you can configure a back side cluster
```

```
#network for OSD object replication, heart beats, backfilling,
#recovery, and so on
public_network = {network}[, {network}]
#cluster_network = {network}[, {network}]

#Clusters require authentication by default.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx

#Choose reasonable numbers for your journals, number of replicas
#and placement groups.
osd_journal_size = {n}
osd_pool_default_size = {n}  # Write an object n times.
osd_pool_default_min_size = {n} # Allow writing n copy in a degraded
state.
osd_pool_default_pg_num = {n}
osd_pool_default_pgp_num = {n}

#Choose a reasonable crush leaf type.
#0 for a 1-node cluster.
#1 for a multi node cluster in a single rack
#2 for a multi node, multi chassis cluster with multiple hosts in a
chassis
#3 for a multi node cluster with hosts across racks, and so on
osd_crush_chooseleaf_type = {n}
```

## 1.3. 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**

## 1.4. VIEWING THE CEPH RUNTIME CONFIGURATION

To view a runtime configuration, log in to a Ceph node and execute:

```
ceph daemon {daemon-type}.{id} config show
```

For example, if you want to see the configuration for **osd.0**, log into the node containing **osd.0** and execute:

```
ceph daemon osd.0 config show
```

For additional options, specify a daemon and **help**. For example:

```
ceph daemon osd.0 help
```

## 1.5. GETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME

To get a specific configuration setting at runtime, log in to a Ceph node and execute:

```
ceph daemon {daemon-type}.{id} config get {parameter}
```

For example to retrieve the public address of **osd.0**, execute:

```
ceph daemon osd.0 config get public_addr
```

## 1.6. SETTING A SPECIFIC CONFIGURATION SETTING AT RUNTIME

There are two general ways to set a runtime configuration:

- by using the Ceph monitor
- by using the administration socket

You can set a Ceph runtime configuration setting by contacting the monitor using the **tell** and **injectargs** command. To use this approach, the monitors and the daemon you are trying to modify must be running:

```
ceph tell {daemon-type}.{daemon id or *} injectargs --{name} {value} [--{name} {value}]
```

Replace **{daemon-type}** with one of **osd** or **mon**. You can apply the runtime setting to all daemons of a particular type with *****, or specify a specific daemon's ID (that is, its number or name). For example, to change the debug logging for a **ceph-osd** daemon named **osd.0** to **0/5**, execute the following command:

```
ceph tell osd.0 injectargs '--debug-osd 0/5'
```

The **tell** command takes multiple arguments, so each argument for **tell** must be within single quotes, and the configuration prepended with two dashes (**'--{config_opt} {opt-val}' ['-{config_opt} {opt-val}']**). Quotes are not necessary for the **daemon** command, because it only takes one argument.

The **ceph tell** command goes through the monitors. If you cannot bind to the monitor, you can still make the change by logging into the host of the daemon whose configuration you want to change using **ceph daemon**. For example:

```
sudo ceph osd.0 config set debug_osd 0/5
```

## 1.7. GENERAL CONFIGURATION REFERENCE

General settings typically get set automatically by deployment tools.

### 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/$cluster/$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 prevents 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`**

## 1.8. MEMORY ALLOCATION

**`TCMalloc`** is an application used for memory allocation. With Red Hat Ceph Storage 1.3.2 and later, it is possible to increase the size of the **`TCMalloc`** thread cache to significantly improve performance of the Ceph cluster.

To set the thread cache size, edit the value of the **`TCMALLOC_MAX_TOTAL_THREAD_CACHE_BYTES`** parameter in Ceph system configuration file. For Ceph clusters running on top of Red Hat Enterprise Linux, modify the **`/etc/sysconfig/ceph`** file. For Ceph clusters running on top of Ubuntu, modify the **`/etc/default/ceph`** file.

Also, with Red Hat Ceph Storage 1.3.2 and later, the default value of **`TCMALLOC_MAX_TOTAL_THREAD_CACHE_BYTES`** has been changed from 32 MB to 128 MB.

# CHAPTER 2. NETWORK CONFIGURATION REFERENCE

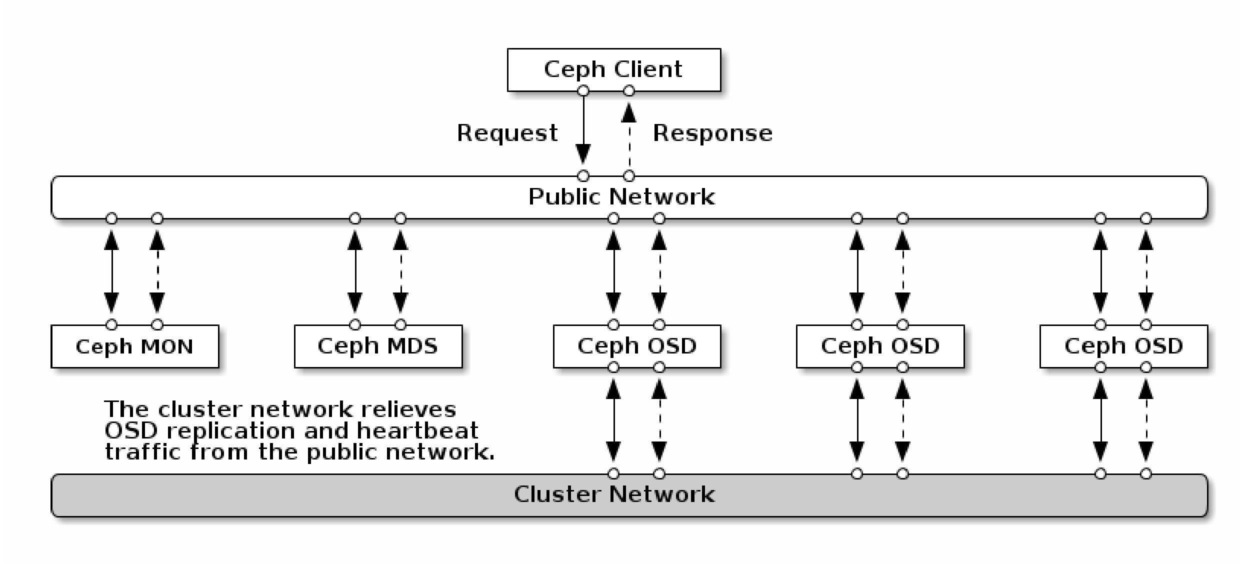
Network configuration is critical for building a high performance Red Hat Ceph Storage cluster. The Ceph storage cluster does not perform request routing or dispatching on behalf of the Ceph client. Instead, Ceph clients make requests directly to Ceph OSD daemons. Ceph OSDs perform data replication on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

All Ceph clusters must use a public network. However, unless you specify a cluster (internal) network, Ceph assumes a single public network. Ceph can function with a public network only, but you will see significant performance improvement with a second "cluster" network in a large cluster.

Red Hat recommends running a Ceph storage cluster with two networks:

- a public network
- and a cluster 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.

## 2.1. NETWORK CONFIGURATION SETTINGS

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`**.

**Tip**

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

**Note**

Ceph uses CIDR notation for subnets (for example, **`10.0.0.0/24`**).

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.

### 2.1.1. Public Network

To configure a public network, add the following option to the **`[global]`** section of the Ceph configuration file.

```
[global]
    ...
    public_network = <public-network/netmask>
```

The public network configuration allows you specifically define IP addresses and subnets for the public network. You may specifically assign static IP addresses or override **`public network`** settings using the **`public addr`** setting for a specific daemon.

**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

### 2.1.2. Cluster Network

If you declare 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. To configure a cluster network, add the following option to the **`[global]`** section of the Ceph configuration file.

```
[global]
    ...
    cluster_network = <cluster-network/netmask>
```

It is preferable, that the cluster network is not reachable from the public network or the Internet for added security.

The cluster network configuration allows you to declare a cluster network, and specifically define IP addresses and subnets for the cluster network. You can specifically assign static IP addresses or override **`cluster network`** settings using the **`cluster addr`** setting for specific OSD daemons.

**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**

N/A

### 2.1.3. Bind

Bind settings set the default port ranges Ceph OSD daemons use. The default range is **`6800:7100`**. Ensure that the firewall configuration allows you to use the configured port range.

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**

**`7100`**

**Required**

No.

**ms_bind_ipv6**

**Description**

Enables Ceph daemons to bind to IPv6 addresses.

**Type**

Boolean

**Default**

**`false`**

**Required**

No

## 2.1.4. Hosts

Ceph expects at least one monitor declared 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.

**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`**

**Tip**

Do not use **`localhost`**. To get your host name, execute the **`hostname -s`** command and use the name of your host to the first period, not the fully-qualified domain name.

**Important**

Do not specify any value for **`host`** when using a third party deployment system that retrieves the host name for you.

### 2.1.5. TCP

Ceph disables TCP buffering by default.

**ms_tcp_nodelay**

**Description**

Ceph enables **`tcp nodelay`** so that each request is sent immediately (no buffering). Disabling Nagle’s algorithm increases network traffic, which can introduce latency. If you experience large numbers of small packets, you may try disabling **`tcp nodelay`**.

**Type**

Boolean

**Required**

No

**Default**

**`true`**

**ms_tcp_rcvbuf**

**Description**

The size of the socket buffer on the receiving end of a network connection. Disable 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.

### 2.1.6. Firewall

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. You can configure this range at your discretion.

```
sudo iptables -L
```

For the **`firewalld`** daemon, execute the following command as **`root`**:

```
# firewall-cmd --list-all-zones
```

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

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

#### 2.1.6.1. Monitor Firewall

Ceph monitors listen on port **`6789`** by default. Additionally, Ceph monitors always operate on the public network. When you add the rule using the example below, make sure you replace **`<iface>`** with the public network interface (for example, **`eth0`**, **`eth1`**, and so on), **`<ip-address>`** with the IP address of the public network and **`<netmask>`** with the netmask for the public network.

```
sudo iptables -A INPUT -i <iface> -p tcp -s <ip-address>/<netmask> --
dport 6789 -j ACCEPT
```

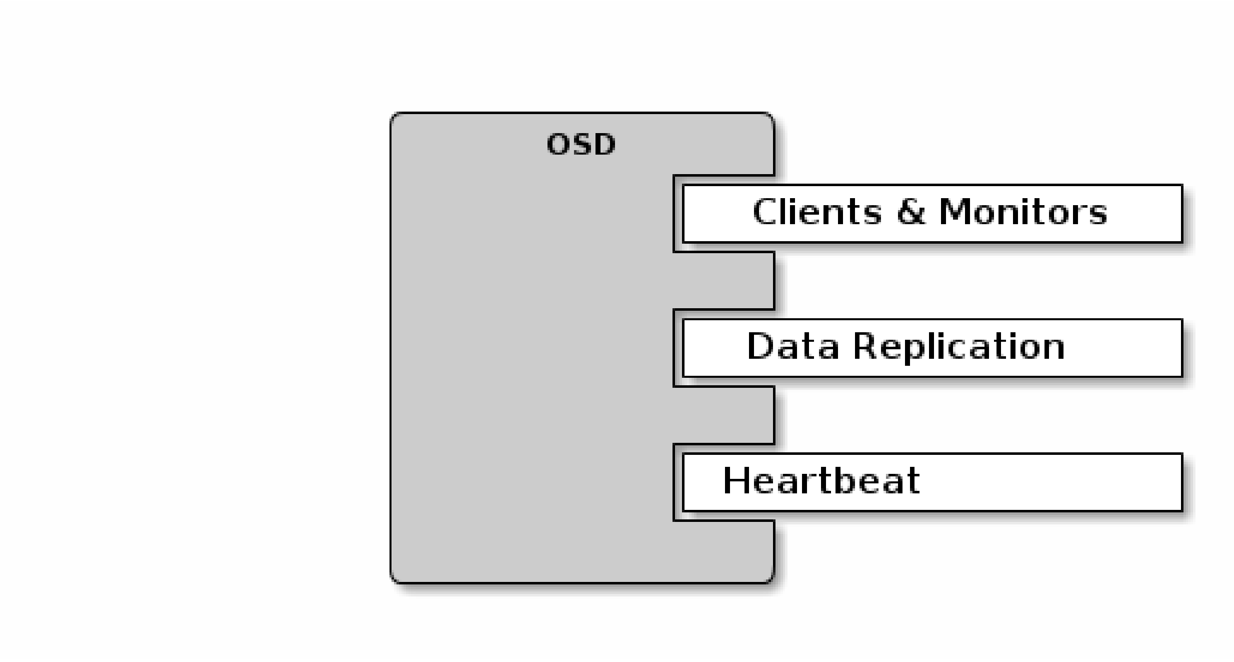
For the **`firewalld`** daemon, execute the following commands as **`root`**:

```
# firewall-cmd --zone=public --add-port=6789/tcp
# firewall-cmd --zone=public --add-port=6789/tcp --permanent
```

#### 2.1.6.2. OSD Firewall

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

1. One for talking to clients and monitors (public network).
2. One for sending data to other OSDs (cluster network).
3. One for sending heartbeat packets (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 to open 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 host.

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.

When you add the rule using the example below, make sure you replace **`<iface>`** with the network interface (for example, **`eth0`** or **`eth1), `<ip-address>`** with the IP address and **`<netmask>`** with the netmask of the public or cluster network. For example:

```
sudo iptables -A INPUT -i <iface>  -m multiport -p tcp -s <ip-address>/<netmask> --dports 6800:6810 -j ACCEPT
```

For the **`firewalld`** daemon, execute the following commands as **`root`**:

```
# firewall-cmd --zone=public --add-port=6800-6810/tcp
# 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.

## 2.2. CEPH DAEMONS

Ceph has one network configuration requirement that applies to all daemons. The Ceph configuration file must specify the **`host`** for each daemon. Ceph no longer requires that a Ceph configuration file specify the monitor IP address and its port.

**Important**

Some deployment utilities might create a configuration file for you. Do not set these values if the deployment utility does it for you.

**Tip**

The **`host`** setting is the short name of the host (that is, not an FQDN). It is not an IP address either. Use the **`hostname -s`** command to retrieve the name of the host.

```
[mon.a]

    host = <hostname>
    mon addr = <ip-address>:6789

[osd.0]
    host = <hostname>
```

You do not have to set the host IP address for a daemon. If you have a static IP configuration and both public and cluster networks running, the Ceph configuration file might specify the IP address of the host for each daemon. To set a static IP address for a daemon, the following option(s) should appear in the daemon instance sections of the Ceph configuration file.

```
[osd.0]
    public_addr = <host-public-ip-address>
    cluster_addr = <host-cluster-ip-address>
```

### One NIC OSD in a Two Network Cluster

Generally, Red Hat does not recommend deploying an OSD host with a single NIC in a cluster with two networks. However, you cam accomplish this by forcing the OSD host to operate on the public network by adding a **`public addr`** entry to the **`[osd.n]`** section of the Ceph configuration file, where **`n`** refers to the number of the OSD with one NIC. Additionally, the public network and cluster network must be able to route traffic to each other, which Red Hat does not recommend for security reasons.

# CHAPTER 3. MONITOR CONFIGURATION REFERENCE

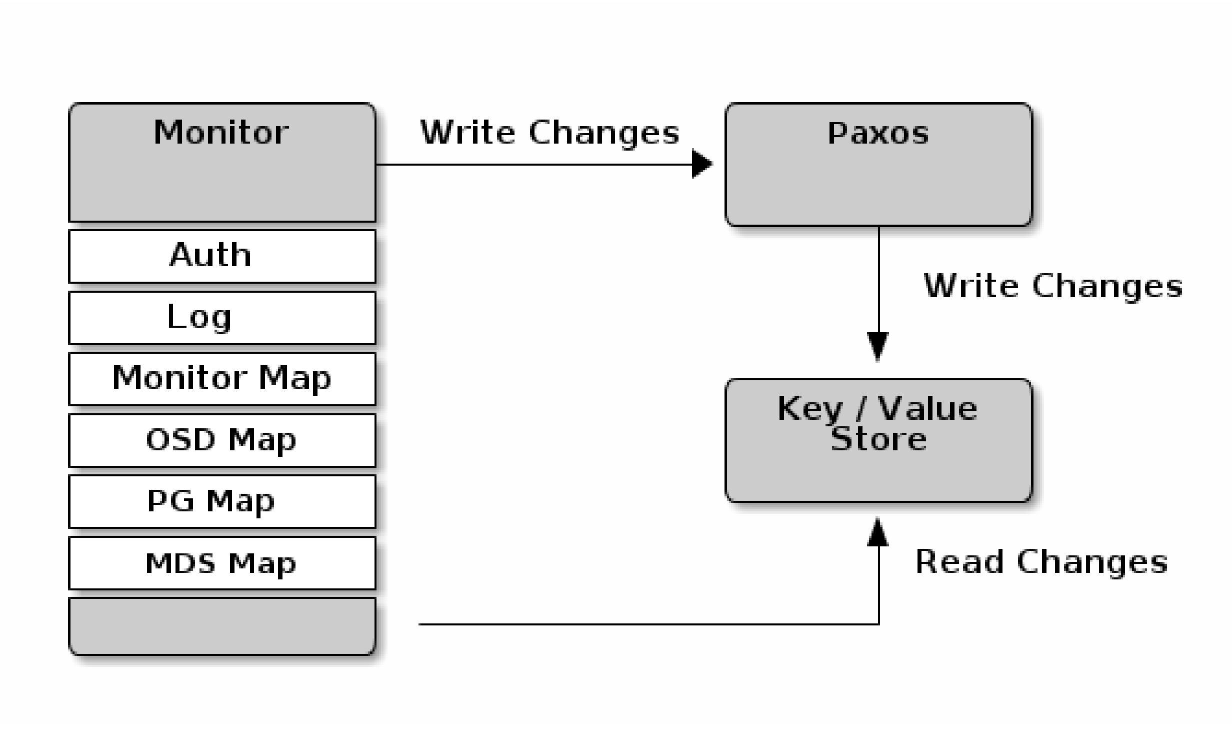
Understanding how to configure a Ceph monitor is an important part of building a reliable Red Hat Ceph Storage cluster. All clusters have at least one monitor. A monitor configuration usually remains fairly consistent, but you can add, remove or replace a monitor in a cluster.

## 3.1. BACKGROUND

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 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 **`leveldb`** database) to perform store-wide synchronization.



### 3.1.1. 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.

### 3.1.2. 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 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).

### 3.1.3. 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 also applies to updates to the monitor map. 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 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.

### 3.1.4. Bootstrapping Monitors

In most configuration and deployment cases, tools that deploy Ceph might help bootstrap the Ceph monitors by generating a monitor map for you (for example, **ceph-deploy**, and so on). 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 clusters on the same hardware, you must specify the unique ID of the object store when bootstrapping a monitor. Deployment tools usually do this for you (for example, **ceph-deploy** can call a tool like **uuidgen**), but you can specify the **fsid** manually too.

- **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. A deployment tool such as **ceph-deploy** usually does this for you, but you can perform this step manually too.

## 3.2. CONFIGURING MONITORS

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 (for example, **[mon.a]**). By convention, monitor instance names use alpha notation.

```
[global]

[mon]

[mon.a]

[mon.b]

[mon.c]
```

### 3.2.1. Minimum Configuration

The bare minimum monitor settings for a Ceph monitor in the Ceph configuration file includes a host name for each monitor. You can configure these under **[mon]** or under the entry for a specific monitor.

```
[mon]
mon host = hostname1,hostname2,hostname3
mon addr = 10.0.0.10:6789,10.0.0.11:6789,10.0.0.12:6789
```

Or

```
[mon.a]
host = hostname1
mon addr = 10.0.0.10:6789
```

**Note**

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

**Important**

Once you deploy a Ceph cluster, do not change the IP address of the monitors.

### 3.2.2. Cluster ID

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.

**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.

**Note**

Do not set this value if you use a deployment tool that does it for you.

### 3.2.3. Initial Members

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 cluster in order to establish a quorum. This may reduce the time it takes for the cluster to come online.

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

**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

**Note**

A *majority* of monitors in your cluster must be able to reach each other in order to establish a quorum. You can decrease the initial number of monitors to establish a quorum with this setting.

### 3.2.4. Data

Ceph provides a default path where Ceph monitors store data. For optimal performance in a production Red Hat Ceph Storage cluster, Red Hat recommends running Ceph monitors on separate hosts and drives from Ceph OSDs. Ceph monitors call the `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.

**Note**

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 `[mon]` section of the configuration file.

**mon_data**

**Description**

The monitor’s data location.

**Type**

String

**Default**

`/var/lib/ceph/mon/$cluster-$id`

### 3.2.5. Storage Capacity

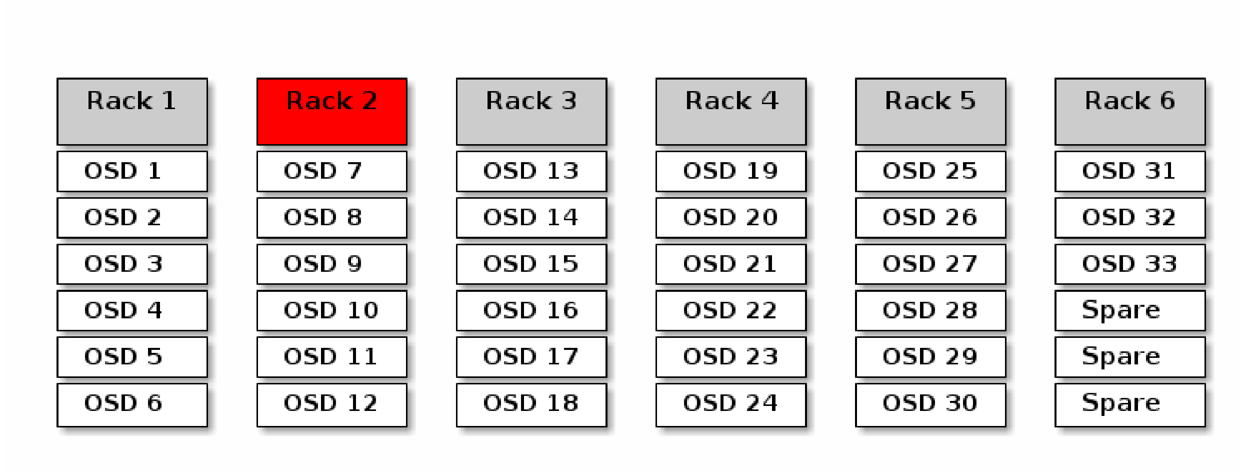
When a Red Hat Ceph Storage cluster gets close to its maximum capacity (specifies by the `mon_osd_full_ratio` parameter), Ceph prevents you from writing to or reading from Ceph OSDs as a safety measure to prevent data loss. Therefore, letting a production Red Hat Ceph Storage cluster approach its full ratio is not a good practice, because it sacrifices high availability. The default full ratio is **`.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 **`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.

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 order to maintain high availability. Ideally, you want to plan for a series of Ceph OSD failures where the cluster can recover to an **`active + clean`** state without replacing those Ceph OSDs immediately. You can run a cluster in an **`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 **`mon osd full ratio`** of **`0.95`**, 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.



| Rack 1 | Rack 2 | Rack 3 | Rack 4 | Rack 5 | Rack 6 |
|---|---|---|---|---|---|
| OSD 1 | OSD 7 | OSD 13 | OSD 19 | OSD 25 | OSD 31 |
| OSD 2 | OSD 8 | OSD 14 | OSD 20 | OSD 26 | OSD 32 |
| OSD 3 | OSD 9 | OSD 15 | OSD 21 | OSD 27 | OSD 33 |
| OSD 4 | OSD 10 | OSD 16 | OSD 22 | OSD 28 | Spare |
| OSD 5 | OSD 11 | OSD 17 | OSD 23 | OSD 29 | Spare |
| OSD 6 | OSD 12 | OSD 18 | OSD 24 | OSD 30 | Spare |

It is normal in such a cluster for one or two OSDs to fail. A less frequent but reasonable scenario involves a rack’s router or power supply failing, which brings down multiple OSDs simultaneously (for example, OSDs 7-12). In such a scenario, you should still strive for a cluster that can remain operational and achieve an **`active + clean`** state, even if that means adding a few hosts with additional OSDs in short order. If your capacity utilization is too high, you might not lose data, but you could still sacrifice data availability while resolving an outage within a failure domain if capacity utilization of the cluster exceeds the full ratio. For this reason, Red Hat recommends at least some rough capacity planning.

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 number of 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.

```
[global]
...
mon_osd_full_ratio = .80
mon_osd_nearfull_ratio = .70
```

**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`**

**Tip**

If some OSDs are **`nearfull`**, but others have plenty of capacity, you might have a problem with the CRUSH weight for the **`nearfull`** OSDs.

### 3.2.6. Heartbeat

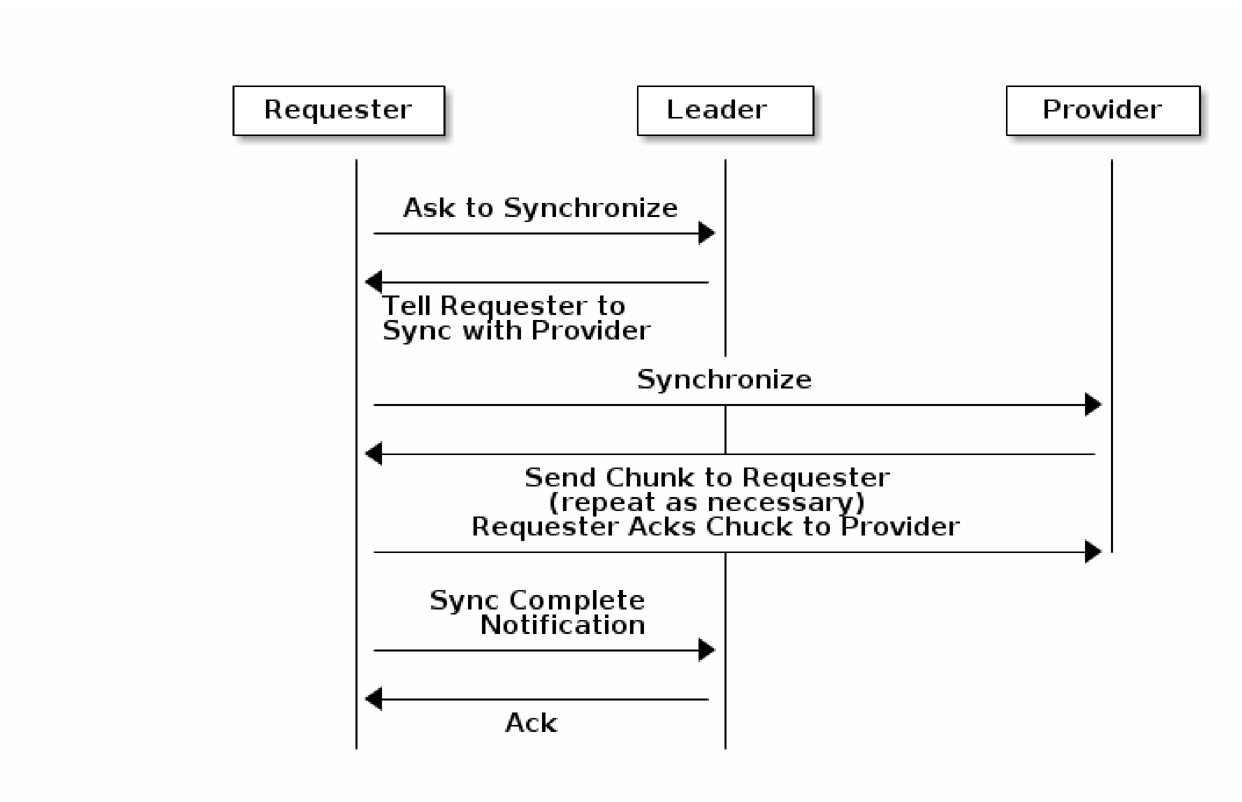
Ceph monitors know about the cluster by requiring reports from each OSD, and by receiving reports from OSDs about the status of their neighboring OSDs. Ceph provides reasonable default settings for interaction between monitor and OSD, however, you can modify them as needed.

### 3.2.7. Monitor Store Synchronization

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. 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 in order 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.


Requester
Leader
Provider
Ask to Synchronize
Tell Requester to Sync with Provider
Synchronize
Send Chunk to Requester (repeat as necessary)
Requester Acks Chuck to Provider
Sync Complete Notification
Ack

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`.

**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**

`30.0`

**mon_sync_timeout**

**Description, 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.

**Type**

32-bit Integer

**Default**

`1045676`

**mon_accept_timeout**

**Description**

Number of seconds the Leader will wait for the Requester(s) to accept a Paxos update. It is also used during the Paxos recovery phase for similar purposes.

**Type**

Float

**Default**

`10.0`

**paxos_propose_interval**

**Description**

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

**Type**

Double

**Default**

`1.0`

**paxos_min_wait**

**Description**

The minimum amount of time to gather updates after a period of inactivity.

**Type**

Double

**Default**

`0.05`

**paxos_trim_tolerance**

**Description**

**Description**

The number of extra proposals tolerated before trimming.

**Type**

Integer

**Default**

`30`

**paxos_trim_disabled_max_versions**

**Description**

The maximum number of version allowed to pass without trimming.

**Type**

Integer

**Default**

`100`

**mon_lease**

**Description**

The length (in seconds) of the lease on the monitor’s versions.

**Type**

Float

**Default**

`5`

**mon_lease_renew_interval**

**Description**

The interval (in seconds) for the Leader to renew the other monitor’s leases.

**Type**

Float

**Default**

`3`

**mon_lease_ack_timeout**

**Description**

The number of seconds the Leader will wait for the Providers to acknowledge the lease extension.

**Type**

Float

**Default**

**`10.0`**

**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`**

### 3.2.8. Clock

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 (for example, timestamps outdated).
- Timeouts triggered too soon or late when a message was not received in time.

See Monitor Store Synchronization for details.

**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 the Monitor Store Synchronization settings may influence the level of acceptable clock drift without compromising Paxos guarantees.

Ceph provides the following tunable options to allow you to find acceptable values.

**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`

### 3.2.9. Client

**mon_client_hunt_interval**

**Description**

The client will try a new monitor every `N` seconds until it establishes a connection.

**Type**

Double

**Default**

`3.0`

**mon_client_ping_interval**

**Description**

The client will ping the monitor every `N` seconds.

**Type**

Double

**Default**

`10.0`

**mon_client_max_log_entries_per_message**

**Description**

The maximum number of log entries a monitor will generate per client message.

**Type**

Integer

**Default**

`1000`

**mon_client_bytes**

**Description**

The amount of client message data allowed in memory (in bytes).

**Type**

64-bit Integer Unsigned

**Default**

`100ul << 20`

## 3.3. MISCELLANEOUS

**mon_max_osd**

**Description**

The maximum number of OSDs allowed in the cluster.

**Type**

32-bit Integer

**Default**

`10000`

**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 file system 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**

**2.0**

**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`

# CHAPTER 4. CEPHX CONFIGURATION REFERENCE

The **cephx** protocol is enabled by default. Cryptographic authentication has some computational costs, though they are generally quite low. If the network environment connecting a client and server hosts is very safe and you cannot afford authentication, you can disable it. However, 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.

## 4.1. DEPLOYMENT SCENARIOS

There are two main scenarios for deploying a Ceph cluster, which impact how you initially configure Cephx. Most first time Ceph users use **ceph-deploy** to create a cluster, which is easiest. For clusters using other deployment tools (for example, Chef, Juju, Puppet, and so on), you will need to use the manual procedures or configure your deployment tool to bootstrap your monitor(s).

### 4.1.1. Using ceph-deploy

When you deploy a cluster with **ceph-deploy**, you do not have to bootstrap the monitor manually or create the **client.admin** user or keyring. The steps you execute in the Storage Cluster Quick Start will invoke **ceph-deploy** to do that for you.

When you execute **ceph-deploy new <initial-monitor(s)>**, Ceph:

- creates a monitor keyring for you (only used to bootstrap monitors)
- generates an initial Ceph configuration file, which contains the following authentication settings, indicating that Ceph enables authentication by default:

```
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

When you execute **ceph-deploy mon create-initial**, Ceph bootstraps the initial monitor(s), retrieve a **ceph.client.admin.keyring** file containing the key for the **client.admin** user. Additionally, it retrieves keyrings that give **ceph-deploy** and **ceph-disk** utilities the ability to prepare and activate OSDs.

When you execute **ceph-deploy admin <node-name>** (**note:** Ceph must be installed first), you push the Ceph configuration file and the **ceph.client.admin.keyring** to the **/etc/ceph/** directory of the node. You will be able to execute Ceph administrative functions as **root** on the command line of that node.

### 4.1.2. Manual

When you deploy a cluster manually, you have to bootstrap the monitor manually and create the **client.admin** user and keyring. To deploy Ceph manually, see our Knowledgebase article. The steps for monitor bootstrapping are the logical steps you must perform when using third party deployment tools like Chef, Puppet, Juju, and so on.

## 4.2. ENABLING AND DISABLING CEPHX

Enabling Cephx requires that you have deployed keys for your monitors and OSDs. If you are simply toggling Cephx on / off, you do not have to repeat the bootstrapping procedures.

### 4.2.1. 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.

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

   ```
   ceph auth get-or-create client.admin mon 'allow *' osd 'allow *'
   -o /etc/ceph/ceph.client.admin.keyring
   ```

   **Warning:** This will erase the contents of any existing **/etc/ceph/client.admin.keyring** file. Do not perform this step if a deployment tool has already done it for you.

2. Create a keyring for the monitor cluster and generate a monitor secret key:

   ```
   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:

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

   ```
   auth_cluster_required = cephx
   auth_service_required = cephx
   auth_client_required = cephx
   ```

6. Start or restart the Ceph cluster.

### 4.2.2. 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. Red Hat recommends enabling authentication. However, it may be easier during setup or troubleshooting to temporarily disable authentication.

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

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

2. Start or restart the Ceph cluster.

## 4.3. CONFIGURATION SETTINGS

### 4.3.1. Enablement

**auth_cluster_required**

**Description**

If enabled, the Red Hat Ceph Storage cluster daemons (that is, **`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**.

### 4.3.2. Keys

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.

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 as **root**:

```
# 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.

**keyring**

**Description**

The path to the keyring file.

**Type**

String

**Required**

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

### 4.3.3. 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, are shown below.

**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'`**

**Note**

The monitor keyring (that is **`mon.`**) contains a key but no capabilities, and is not part of the cluster **`auth`** database.

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

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

For example, **`osd.12`** is:

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

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

### 4.3.4. Signatures

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

Like other parts of Ceph authentication, Ceph provides fine-grained control so you can enable or disable signatures for service messages between the client and Ceph, and you can enable or disable signatures for messages between Ceph daemons.

**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`

**Note**

Ceph kernel modules do not support signatures yet.

### 4.3.5. Time to Live

**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`

# CHAPTER 5. POOL, PG, AND CRUSH CONFIGURATION REFERENCE

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. 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.

```
[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
```

## 5.1. SETTINGS

**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`**

**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_erasure_code_stripe_width**

**Description**

Sets the desired size, in bytes, of an object stripe on every erasure coded pools. Every object if size S will be stored as N stripes and each stripe will be encoded/decoded individually.

**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 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 (for now).

**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_pgls**

**Description**

**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`

# CHAPTER 6. OSD CONFIGURATION REFERENCE

You can configure Ceph OSDs in the Ceph configuration file, but Ceph OSDs can use the default values and a very minimal configuration. A minimal Ceph OSD configuration sets the **`osd journal size`** and **`osd host`** options, and uses default values for almost everything else.

Ceph OSDs are numerically identified in incremental fashion, beginning with **`0`** using the following convention:

```
osd.0
osd.1
osd.2
```

In a configuration file, you can specify settings for all Ceph OSDs in the cluster by adding configuration settings to the **`[osd]`** section of the configuration file. To add settings directly to a particular Ceph OSD (for example, **`osd host`**), enter it in a section specific only to that OSD in the Ceph configuration file. For example:

```
[osd]
osd journal size = 1024

[osd.0]
osd host = osd-host-a

[osd.1]
osd host = osd-host-b
```

## 6.1. GENERAL SETTINGS

The following settings provide a Ceph OSD's ID, and determine paths to data and journals. Ceph deployment scripts typically generate the UUID automatically.

**Important**

Red Hat does not recommend changing the default paths for data or journals, as it makes it more problematic to troubleshoot Ceph later.

The journal size should be at least twice the product of the expected drive speed multiplied by the value of the **`filestore max sync interval`** option. However, the most common practice is to partition the journal drive (often an SSD), and mount it such that Ceph uses the entire partition for the journal.

### 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. 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`**

## 6.2. JOURNAL SETTINGS

By default, Ceph expects that you will store a Ceph OSD’s journal with the following path:

```
/var/lib/ceph/osd/$cluster-$id/journal
```

Without performance optimization, Ceph stores the journal on the same disk as the Ceph OSD’s data. A Ceph OSD optimized for performance can use a separate disk to store journal data, for example, a solid state drive delivers high performance journaling.

A journal size should find the product of the **`filestore max sync interval`** and the expected throughput, and multiply the product by two (2):

```
osd journal size = <2 * (expected throughput * filestore max sync
interval)>
```

The expected throughput number should include the expected disk throughput (that is, sustained data transfer rate), and network throughput. For example, a 7200 RPM disk will likely have approximately 100 MB/s. Taking the **`min()`** of the disk and network throughput should provide a reasonable expected throughput. Some users just start off with a 10GB journal size. For example:

```
osd journal size = 10000
```

**Warning**

Sizing the journal correctly for your OSDs is important. Using a small journal will lead to a slower recovery in the event of an OSD failure. The number of recovery threads has to be decreased in order to have a stable recovery by keeping pressure in journal at an acceptable level. Also, committing transactions to the file store will be slower and could lead to the file store hanging if the queued transaction size is bigger than the journal size.

### osd journal

**Description**

The path to the OSD’s journal. This may be a path to a file or a block device (such as a partition of an SSD). If it is a file, you must create the directory to contain it. We recommend using a drive separate from the **`osd data`** drive.

**Type**

String

**Default**

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

#### osd journal size

**Description**

The size of the journal in megabytes. If this is 0, and the journal is a block device, the entire block device is used. This is ignored if the journal is a block device, and the entire block device is used.

**Type**

32-bit Integer

**Default**

**`5120`**

**Recommended**

Begin with 1GB. Should be at least twice the product of the expected speed multiplied by **`filestore max sync interval`**.

## 6.3. SCRUBBING

In addition to making multiple copies of objects, Ceph insures 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.

#### 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 maximum 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 ration and randomizes the scheduled scrub between **`osd scrub min interval`** and **`osd scrub max interval`**.

**Type**

Float

**Default**

**`0.5`**.

**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 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 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`

## 6.4. OPERATIONS

Operations settings allow you to configure the number of threads for servicing requests. If you set the `osd op threads` parameter to `0`, it disables multi-threading.

By default, Ceph uses two threads with a 30 second timeout and a 30 second complaint time if an operation does not complete within those time parameters. Set operations priority weights between client operations and recovery operations to ensure optimal performance during recovery.

**osd op threads**

**Description**

The number of threads to service Ceph OSD operations. Set to `0` to disable it. Increasing the number might increase the request processing rate.

**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**

**`10`**

**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**

an empty string

### 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**

**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.

## 6.5. BACKFILLING

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.

**osd max backfills**

**Description**

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

**Type**

64-bit Unsigned Integer

**Default**

`10`

**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**

`0.85`

**osd backfill retry interval**

**Description**

The number of seconds to wait before retrying backfill requests.

**Type**

Double

**Default**

`10.0`

## 6.6. OSD MAP

OSD maps reflect the OSD daemons operating in the cluster. Over time, the number of map epochs increases. Ceph provides the following settings to ensure that Ceph performs well as the OSD map grows larger.

**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**

`500`

**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**

`100`

## 6.7. RECOVERY

When the cluster starts or when a Ceph OSD terminates unexpectedly and restarts, the OSD begins peering with other Ceph OSDs before 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 may 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 recovery requests, threads and object chunk sizes which allows Ceph perform well in a degraded state.

**osd recovery delay start**

**Description**

After peering completes, Ceph will delay 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 places an increased load on the cluster.

**Type**

32-bit Integer

**Default**

`15`

**osd recovery max chunk**

**Description**

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

**Type**

64-bit Integer Unsigned

**Default**

`8 << 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`

## 6.8. MISCELLANEOUS

**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 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**

**`256`**

**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`**

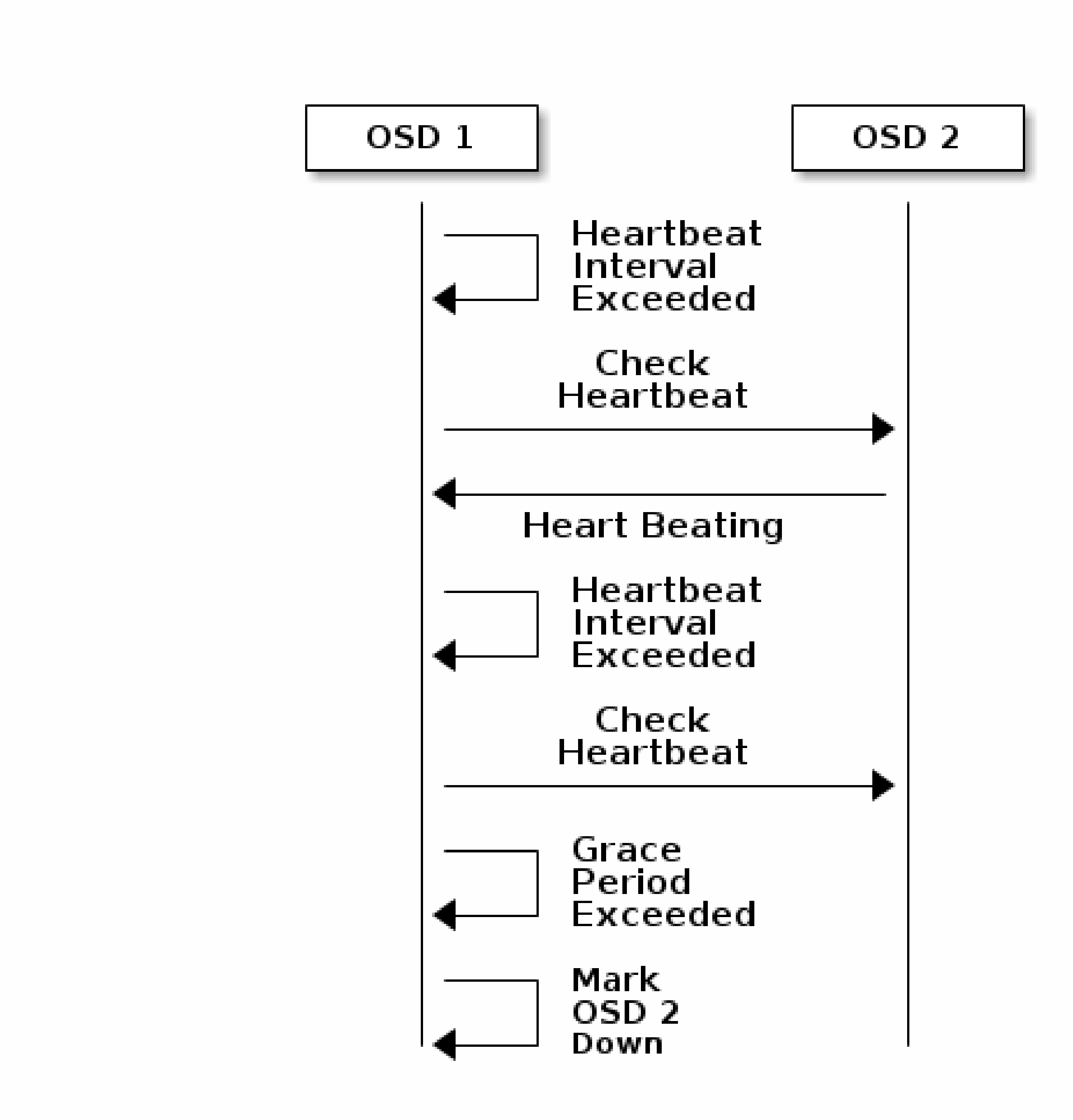
# CHAPTER 7. CONFIGURING MONITOR AND OSD INTERACTION

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 Ceph OSD Daemon 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.

## 7.1. OSDS CHECK HEARTBEATS

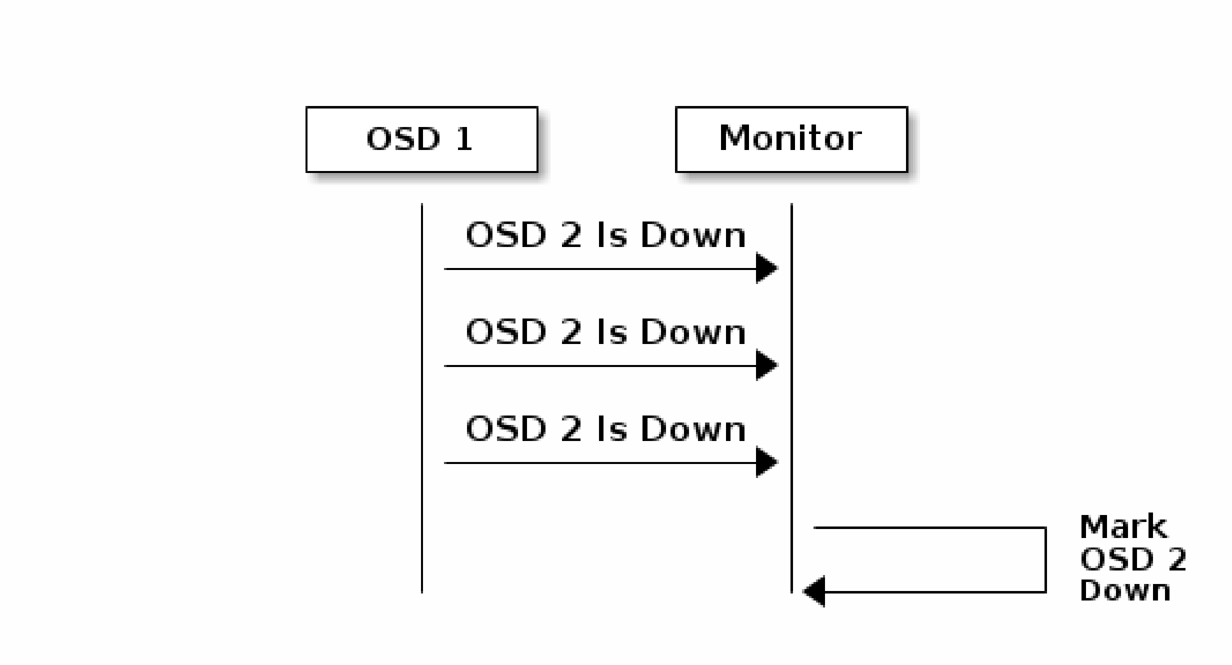
Each Ceph OSD Daemon checks the heartbeat of other Ceph OSD Daemons every 6 seconds. You can change the heartbeat interval by adding an **`osd heartbeat interval`** setting under the **`[osd]`** section of your Ceph configuration file, or by setting the value at runtime. If a neighboring Ceph OSD Daemon doesn’t show a heartbeat within a 20 second grace period, the Ceph OSD Daemon may consider the neighboring Ceph OSD Daemon **down** and report it back to a Ceph Monitor, which will update the Ceph Cluster Map. You may change this grace period by adding an **`osd heartbeat grace`** setting under the **`[osd]`** section of your Ceph configuration file, or by setting the value at runtime.



## 7.2. OSDS REPORT DOWN OSDS

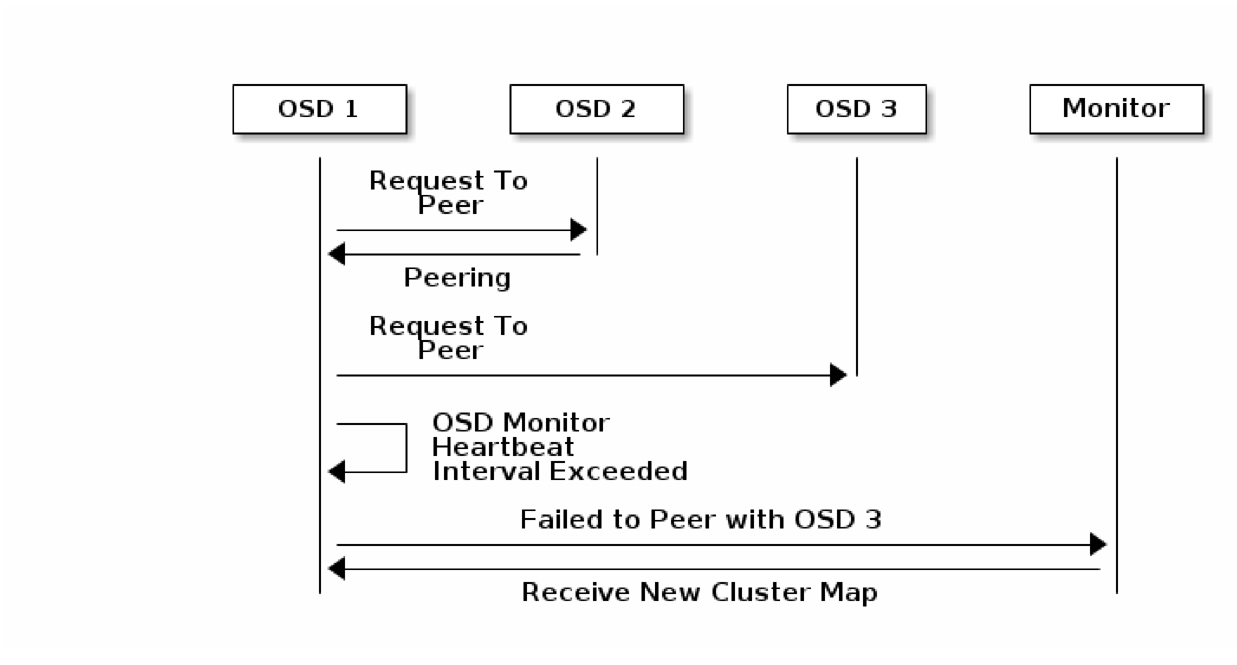
By default, a Ceph OSD Daemon must report to the Ceph Monitors that another Ceph OSD Daemon is **down** three times before the Ceph Monitors acknowledge that the reported Ceph OSD Daemon is **down**. You can change the minimum number of **`osd down`** reports by adding the **`mon osd min down reports`** setting under the **`[mon]`** section of the Ceph configuration file, or by setting the value at runtime.

By default, only one Ceph OSD Daemon is required to report another Ceph OSD Daemon **down**. You can change the number of Ceph OSD Daemons required to report a Ceph OSD Daemon **down** to a Ceph Monitor by adding the **`mon osd min down reporters`** setting under the **`[mon]`** section of the Ceph configuration file, or by setting the value at runtime.



## 7.3. OSDS REPORT 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 will ping a Ceph Monitor for the most recent copy of the cluster map every 30 seconds. You can change the Ceph Monitor heartbeat interval by adding the **`osd mon heartbeat interval`** setting under the **`[osd]`** section of the Ceph configuration file, or by setting the value at runtime.

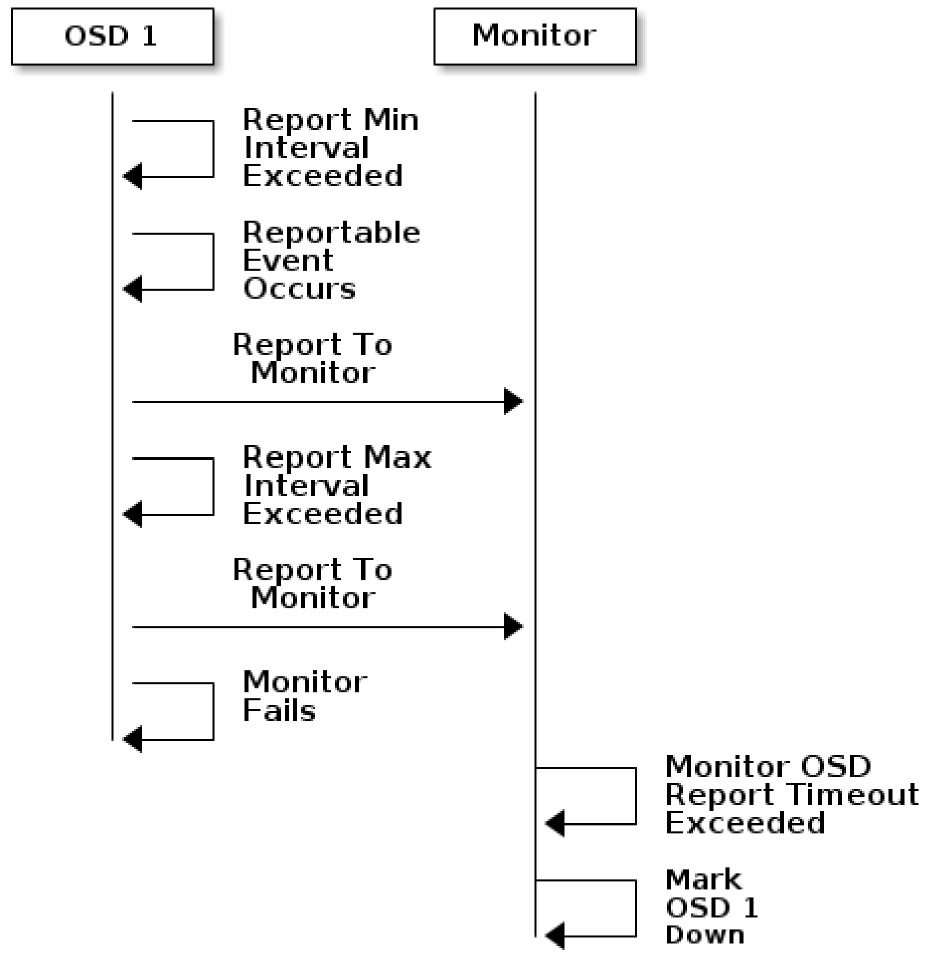


## 7.4. OSDS REPORT THEIR STATUS

If an Ceph OSD Daemon does not report to a Ceph Monitor, the Ceph Monitor will consider the Ceph OSD Daemon **down** after the **`mon osd report timeout`** 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 adding the **`osd mon report interval min`** setting under the **`[osd]`** section of the Ceph configuration file, or by setting the value at runtime.

A Ceph OSD Daemon sends a report to a Ceph Monitor every 120 seconds irrespective of whether any notable changes occur. You can change the Ceph Monitor report interval by adding the **`osd mon report interval max`** setting under the **`[osd]`** section of the Ceph configuration file, or by setting the value at runtime.



## 7.5. CONFIGURATION SETTINGS

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

### 7.5.1. Monitor Settings

**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_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 scaled 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**

**`300`**

**mon_osd_downout_subtree_limit**

**Description**

The largest CRUSH unit type that Ceph will automatically mark **`out`**.

**Type**

String

**Default**

**`rack`**

**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**

**`1`**

**mon_osd_min_down_reports**

**Description**

The minimum number of times a Ceph OSD Daemon must report that another Ceph OSD Daemon is **down**.

**Type**

32-bit Integer

**Default**

**3**

### 7.5.2. OSD Settings

**osd_heartbeat_address**

**Description**

An Ceph OSD Daemon’s network address for heartbeats.

**Type**

Address

**Default**

The host address.

**osd_heartbeat_interval**

**Description**

How often an Ceph OSD Daemon pings its peers (in seconds).

**Type**

32-bit Integer

**Default**

**6**

**osd_heartbeat_grace**

**Description**

The elapsed time when a Ceph OSD Daemon has not shown a heartbeat that the Ceph Storage Cluster considers it **down**.

**Type**

32-bit Integer

**Default**

**20**

**osd_mon_heartbeat_interval**

**Description**

How often the Ceph OSD Daemon pings a Ceph Monitor if it has no Ceph OSD Daemon peers.

**Type**

32-bit Integer

**Default**

`30`

**osd_mon_report_interval_max**

**Description**

The maximum time in seconds that a Ceph OSD Daemon can wait before it must report to a Ceph Monitor.

**Type**

32-bit Integer

**Default**

`120`

**osd_mon_report_interval_min**

**Description**

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`

# CHAPTER 8. FILE STORE CONFIGURATION REFERENCE

## 8.1. EXTENDED ATTRIBUTES

Extended attributes (XATTRs) are an important aspect in the CephFS configuration. Some file systems have limits on the number of bytes stored in extended attributes. Additionally, in some cases, the file system might not be as fast as an alternative method of storing extended attributes. The following settings improve CephFS performance by using a method of storing extended attributes that is extrinsic to the underlying file system.

Ceph extended attributes are stored as **`inline xattr`**, using the extended attributes provided by the underlying file system, if it does not impose a size limit. If there is a size limit (4KB total on ext4, for instance), some Ceph extended attributes will be stored in an key-value database called **`omap`** when the **`filestore max inline xattr size`** or **`filestore max inline xattrs`** threshold are reached.

### filestore_xattr_use_omap

**Description**

Use object map for XATTRS. Set to **`true`** for ext4 file systems.

**Type**

Boolean

**Required**

No

**Default**

**`false`**

### filestore_omap_header_cache_size

**Description**

Determines the size of the LRU used to cache object **`omap`** headers. Larger values use more memory but can reduce lookups on **`omap`**. (Experts only).

**Type**

Integer

**Default**

**`1024`**

### filestore_omap_backend

**Description**

Used to determine which back end is used for the **`omap`**. Can be set to **`leveldb`** or **`rocksdb`**. (Experts only. **`rocksdb`** is experimental.)

**Type**

String

**Default**

`leveldb`

**filestore_debug_omap_check**

**Description**

Debugging check on synchronization. Expensive. For debugging only.

**Type**

Boolean

**Required**

No

**Default**

`0`

**filestore_max_inline_xattr_size**

**Description**

The maximum size of an extended attribute stored in a file system (that is, XFS, btrfs, ext4, and others) per object. Should not be larger than the file system can handle.

**Type**

Unsigned 32-bit Integer

**Required**

No

**Default**

`512`

**filestore_max_inline_xattrs**

**Description**

The maximum number of extended attributes stored in the file system per object.

**Type**

32-bit Integer

**Required**

No

**Default**

**2**

### filestore_max_inline_xattr_size_xfs

**Description**

The maximum size of an extended attribute stored in the file system for XFS file systems per object. Should not be larger than the file system can handle.

**Type**

Unsigned 32-bit Integer

**Default**

`65536`

### filestore_max_inline_xattr_size_btrfs

**Description**

The maximum size of an extended attribute stored in the file system for btrfs per object. Should not be larger than the file system can handle.

**Type**

Unsigned 32-bit Integer

**Default**

`2048`

### filestore_max_inline_xattr_size_other

**Description**

The maximum size of an extended attribute stored in the file system for file systems other than btrfs or XFS per object. Should not be larger than the file system can handle.

**Type**

Unsigned 32-bit Integer

**Default**

`512`

### filestore_max_inline_xattrs

**Description**

The maximum number of extended attributes stored in the file system per object. Overrides fine-grained settings.

**Type**

Unsigned 32-bit Integer

**Default**

**0**

**filestore_max_inline_xattrs_xfs**

**Description**

The maximum number of extended attributes stored in an XFS file system per object.

**Type**

Unsigned 32-bit Integer

**Default**

**10**

**filestore_max_inline_xattrs_btrfs**

**Description**

The maximum number of extended attributes stored in a btrfs file system per object.

**Type**

Unsigned 32-bit Integer

**Default**

**10**

**filestore_max_inline_xattrs_other**

**Description**

The maximum number of extended attributes stored in file systems other than btrfs or XFS per object.

**Type**

Unsigned 32-bit Integer

**Default**

**2**

## 8.2. SYNCHRONIZATION INTERVALS

Periodically, the file store needs to quiesce write operations and synchronize the file system, which creates a consistent commit point. It can then free journal entries up to the commit point. Synchronizing more frequently tends to reduce the time required to perform synchronization, and reduces the amount of data that needs to remain in the journal. Less frequent synchronization allows the backing file system to coalesce small writes and metadata updates more optimally—potentially resulting in more efficient synchronization.

**filestore_max_sync_interval**

**Description**

The maximum interval in seconds for synchronizing the file store.

**Type**

Double

**Required**

No

**Default**

**`5`**

**filestore_min_sync_interval**

**Description**

The minimum interval in seconds for synchronizing the file store.

**Type**

Double

**Required**

No

**Default**

**`.01`**

## 8.3. FLUSHER

The file store flusher forces data from large write operations to be written out using the **`sync file range`** option before the synchronization in order to reduce the cost of the eventual synchronization. In practice, disabling the file store flusher seems to improve performance in some cases.

**filestore_flusher**

**Description**

Enables the file store flusher.

**Type**

Boolean

**Required**

No

**Default**

**`false`**

**filestore_flusher_max_fds**

**Description**

Sets the maximum number of file descriptors for the flusher.

**Type**

Integer

**Required**

No

**Default**

`512`

**filestore_sync_flush**

**Description**

Enables the synchronization flusher.

**Type**

Boolean

**Required**

No

**Default**

`false`

**filestore_fsync_flushes_journal_data**

**Description**

Flush journal data during file system synchronization.

**Type**

Boolean

**Required**

No

**Default**

`false`

## 8.4. QUEUE

The following settings provide limits on the size of the file store queue.

**filestore_queue_max_ops**

**Description**

Defines the maximum number of operations in progress that the file store accepts before blocking on queuing new operations.

**Type**

Integer

**Required**

No. Minimal impact on performance.

**Default**

**`500`**

**filestore_queue_max_bytes**

**Description**

The maximum number of bytes for an operation.

**Type**

Integer

**Required**

No

**Default**

**`100 << 20`**

**filestore_queue_committing_max_ops**

**Description**

The maximum number of operations that the file store can commit.

**Type**

Integer

**Required**

No

**Default**

**`500`**

**filestore_queue_committing_max_bytes**

**Description**

The maximum number of bytes that the file store can commit.

**Type**

Integer

**Required**

No

**Default**

**`100 << 20`**

## 8.5. WRITEBACK THROTTLE

Ceph replicates some of the write-back behavior in the kernel, because the page cache tends to keep dirty data round too long.

### filestore_wbthrottle_enable

**Description**

Enables the file store write-back throttle. The file store write-back throttle is used to prevent large amounts of uncommitted data from building up before each file store sync. (Experts only).

**Type**

Boolean

**Default**

**`true`**

### filestore_wbthrottle_btrfs_bytes_start_flusher

**Description**

Dirty bytes threshold at which Ceph begins background flushing for the btrfs file system.

**Type**

64-bit Unsigned Integer

**Default**

**`41943040`**

### filestore_wbthrottle_btrfs_bytes_hard_limit

**Description**

Dirty bytes threshold at which Ceph begins to throttle I/O until the flusher catches up for btrfs.

**Type**

64-bit Unsigned Integer

**Default**

**`419430400`**

**filestore_wbthrottle_btrfs_ios_start_flusher**

**Description**

Dirty I/Os threshold at which Ceph begins background flushing for btrfs.

**Type**

64-bit Unsigned Integer

**Default**

`500`

**filestore_wbthrottle_btrfs_ios_hard_limit**

**Description**

Dirty I/Os threshold at which Ceph begins to throttle IO until the flusher catches up for btrfs.

**Type**

64-bit Unsigned Integer

**Default**

`5000`

**filestore_wbthrottle_btrfs_inodes_start_flusher**

**Description**

Dirty inodes threshold at which Ceph begins background flushing for btrfs.

**Type**

64-bit Unsigned Integer

**Default**

`500`

**filestore_wbthrottle_btrfs_inodes_hard_limit**

**Description**

Dirty inodes threshold at which Ceph begins to throttle IO until the flusher catches up for btrfs. Must be less than the `fd` limit.

**Type**

64-bit Unsigned Integer

**Default**

`5000`

**filestore_wbthrottle_xfs_bytes_start_flusher**

**Description**

Dirty bytes threshold at which Ceph begins background flushing for the XFS file system.

**Type**

64-bit Unsigned Integer

**Default**

**`41943040`**

**filestore_wbthrottle_xfs_bytes_hard_limit**

**Description**

Dirty bytes threshold at which Ceph begins to throttle IO until the flusher catches up for XFS.

**Type**

64-bit Unsigned Integer

**Default**

**`419430400`**

**filestore_wbthrottle_xfs_ios_start_flusher**

**Description**

Dirty I/Os threshold at which Ceph begins background flushing for XFS.

**Type**

64-bit Unsigned Integer

**Default**

**`500`**

**filestore_wbthrottle_xfs_ios_hard_limit**

**Description**

Dirty I/Os threshold at which Ceph begins to throttle IO until the flusher catches up for XFS.

**Type**

64-bit Unsigned Integer

**Default**

**`5000`**

**filestore_wbthrottle_xfs_inodes_start_flusher**

**Description**

**Description**

Dirty inodes threshold at which Ceph begins background flushing for XFS.

**Type**

64-bit Unsigned Integer

**Default**

**`500`**

### filestore_wbthrottle_xfs_inodes_hard_limit

**Description**

Dirty inodes threshold at which Ceph begins to throttle IO until the flusher catches up for XFS. Must be less than the **`fd`** limit.

**Type**

64-bit Unsigned Integer

**Default**

**`5000`**

## 8.6. TIMEOUTS

### filestore_op_threads

**Description**

The number of file system operation threads that execute in parallel.

**Type**

Integer

**Required**

No

**Default**

**`2`**

### filestore_op_thread_timeout

**Description**

The timeout for a file system operation thread (in seconds).

**Type**

Integer

**Required**

No

**Default**

`60`

**filestore_op_thread_suicide_timeout**

**Description**

The timeout for a commit operation before canceling the commit (in seconds).

**Type**

Integer

**Required**

No

**Default**

`180`

## 8.7. B-TREE FILE SYSTEM

**filestore_btrfs_snap**

**Description**

Enable snapshots for a btrfs file store.

**Type**

Boolean

**Required**

No. Only used for btrfs.

**Default**

`true`

**filestore_btrfs_clone_range**

**Description**

Enable cloning ranges for a btrfs file store.

**Type**

Boolean

**Required**

No. Only used for btrfs.

**Default**

`true`

## 8.8. JOURNAL

### filestore_journal_parallel

**Description**

Enables parallel journaling, default for btrfs.

**Type**

Boolean

**Required**

No

**Default**

`false`

### filestore_journal_writeahead

**Description**

Enables write-ahead journaling, default for XFS.

**Type**

Boolean

**Required**

No

**Default**

`false`

### filestore_journal_trailing

**Description**

Deprecated, never use.

**Type**

Boolean

**Required**

No

**Default**

`false`

## 8.9. MISCELLANEOUS

**filestore_merge_threshold**

**Description**

Minimum number of files in a subdirectory before merging into parent NOTE: A negative value means to disable subdirectory merging.

**Type**

Integer

**Required**

No

**Default**

`10`

**filestore_split_multiple**

**Description**

`filestore_split_multiple * abs(filestore_merge_threshold) * 16` is the maximum number of files in a subdirectory before splitting into child directories.

**Type**

Integer

**Required**

No

**Default**

`2`

**filestore_update_to**

**Description**

Limits file store auto upgrade to specified version.

**Type**

Integer

**Required**

No

**Default**

`1000`

**filestore_blackhole**

**Description**

Drop any new transactions on the floor.

**Type**

Boolean

**Required**

No

**Default**

`false`

**filestore_dump_file**

**Description**

File onto which store transaction dumps.

**Type**

Boolean

**Required**

No

**Default**

`false`

**filestore_kill_at**

**Description**

Inject a failure at the n’th opportunity.

**Type**

String

**Required**

No

**Default**

`false`

**filestore_fail_eio**

**Description**

Fail or terminate unexpectedly on EIO.

**Type**

Boolean

**Required**

No

**Default**

`true`

# CHAPTER 9. JOURNAL CONFIGURATION REFERENCE

Ceph OSDs use a journal for the following reasons:

**Speed**

The journal enables the Ceph OSD Daemon to commit small write operations quickly. Ceph writes small, random I/O to the journal sequentially, which tends to speed up bursty workloads by allowing the backing file system more time to coalesce write operations. The Ceph OSD Daemon's journal, however, can lead to spiky performance with short spurts of high-speed writes followed by periods without any write progress as the file system catches up to the journal.

**Consistency**

Ceph OSD Daemons require a file system interface that guarantees atomic compound operations. Ceph OSD Daemons write a description of the operation to the journal and apply the operation to the file system. This enables atomic updates to an object (for example, placement group metadata). Every few seconds—between **`filestore max sync interval`** and **`filestore min sync interval`** settings—the Ceph OSD stops write operations and synchronizes the journal with the file system, allowing Ceph OSDs to trim operations from the journal and reuse the space. On failure, Ceph OSDs replay the journal starting after the last synchronization operation.

## 9.1. SETTINGS

Ceph OSD Daemons support the following journal settings:

**journal_dio**

**Description**

Enables direct I/O to the journal. Requires the **`journal block align`** option set to **`true`**.

**Type**

Boolean

**Required**

Yes when using **`aio`**.

**Default**

**`true`**

**journal_aio**

**Description**

Enables using **`libaio`** for asynchronous writes to the journal. Requires the **`journal dio`** option set to **`true`**.

**Type**

Boolean

**Required**

No.

**Default**

**`true`**.

**journal_block_align**

**Description**

Block aligns write operations. Required for **`dio`** and **`aio`**.

**Type**

Boolean

**Required**

Yes when using **`dio`** and **`aio`**.

**Default**

**`true`**

**journal_max_write_bytes**

**Description**

The maximum number of bytes the journal will write at any one time.

**Type**

Integer

**Required**

No

**Default**

**`10 << 20`**

**journal_max_write_entries**

**Description**

The maximum number of entries the journal will write at any one time.

**Type**

Integer

**Required**

No

**Default**

**100**

**journal_queue_max_ops**

**Description**

The maximum number of operations allowed in the queue at any one time.

**Type**

Integer

**Required**

No

**Default**

**500**

**journal_queue_max_bytes**

**Description**

The maximum number of bytes allowed in the queue at any one time.

**Type**

Integer

**Required**

No

**Default**

**10 << 20**

**journal_align_min_size**

**Description**

Align data payloads greater than the specified minimum.

**Type**

Integer

**Required**

No

**Default**

**64 << 10**

**journal_zero_on_create**

**Description**

Causes the file store to overwrite the entire journal with **`0’s during `mkfs`**.

**Type**

Boolean

**Required**

No

**Default**

**`false`**

# CHAPTER 10. LOGGING AND DEBUGGING

Usually, you add debugging to the Ceph configuration at runtime. You can also add Ceph debug logging to the Ceph configuration file if you are encountering issues when starting your cluster. Also, view Ceph log files under **/var/log/ceph**.

**Tip**

When debug output slows down your system, the latency can hide race conditions.

Logging is resource intensive. If you are encountering a problem in a specific area of your cluster, enable logging for that area of the cluster. For example, if your OSDs are running fine, but your gateways are not, you should start by enabling debug logging for the specific gateway instance(s) giving you trouble. Enable logging for each subsystem as needed.

**Important**

Verbose logging can generate over 1GB of data per hour. If your OS disk reaches its capacity, the node will stop working.

If you enable or increase the rate of Ceph logging, ensure that you have sufficient disk space on your OS disk. See Accelerating Log Rotation for details on rotating log files. When your system is running well, remove unnecessary debugging settings to ensure your cluster runs optimally. Logging debug output messages is relatively slow, and a waste of resources when operating your cluster.

See Subsystem, Log and Debug Settings for details on available settings.

## 10.1. RUNTIME

To see the configuration settings at runtime, log in to a host with a running daemon and execute the following command:

```
ceph --admin-daemon </path/to/admin/socket> config show | less
ceph --admin-daemon /var/run/ceph/ceph-osd.0.asok config show | less
```

To activate Ceph's debugging output (**dout()**) at runtime, use the **ceph tell** command to inject arguments into the runtime configuration:

```
ceph tell <daemon-type>.<daemon id or *> injectargs --<name> <value> [-
-<name> <value>]
```

Replace **<daemon-type>** with one of **osd** or **mon**. To apply the runtime setting to all daemons of a particular type, use *, or specify a specific daemon's ID (that is, its number or letter). For example, to increase debug logging for a **ceph-osd** daemon named **osd.0**:

```
ceph tell osd.0 injectargs --debug-osd 0/5
```

The **ceph tell** command goes through the monitors. If you cannot bind to the monitor, you can still make the change by logging into the host of the daemon whose configuration you want to change by using the **ceph --admin-daemon** command. For example:

```
sudo ceph --admin-daemon /var/run/ceph/ceph-osd.0.asok config set
debug_osd 0/5
```

See Subsystem, Log and Debug Settings for details on available settings.

## 10.2. BOOT TIME

To activate Ceph’s debugging output (**`dout()`**) at boot time, add the debug settings to the Ceph configuration file. Subsystems common to each daemon can be set under the **`[global]`** section in the configuration file. Subsystems for particular daemons are set under the daemon section in the configuration file (that is, **`[mon]`**, **`[osd]`**). For example:

```
[global]
debug_ms = 1/5

[mon]
debug_mon = 20
debug_paxos = 1/5
debug_auth = 2

[osd]
debug_osd = 1/5
debug_filestore = 1/5
debug_journal = 1
debug_monc = 5/20
```

See Subsystem, Log and Debug Settings for details on available settings.

## 10.3. ACCELERATING LOG ROTATION

If your OS disk is relatively full, you can accelerate log rotation by modifying the Ceph log rotation file at **`/etc/logrotate.d/ceph`**. Add a size setting after the rotation frequency to accelerate log rotation by using the **`Cron`** utility if your logs exceed the size setting. For example, the default setting looks like this:

```
rotate 7
weekly
compress
sharedscripts
```

Modify the configuration by adding the **`size`** setting.

```
rotate 7
weekly
size 500M
compress
sharedscripts
```

Then, start the **`crontab`** editor for your user space.

```
crontab -e
```

Finally, add an entry to check the **etc/logrotate.d/ceph** file.

```
30 * * * * /usr/sbin/logrotate /etc/logrotate.d/ceph >/dev/null 2>&1
```

The preceding example checks the **etc/logrotate.d/ceph** file every 30 minutes.

## 10.4. VALGRIND

Debugging might also require you to track down memory and threading issues. You can run a single daemon, a type of daemon, or the whole cluster with the **Valgrind** utility. You should only use **Valgrind** when developing or debugging Ceph. **Valgrind** is computationally expensive, and will slow down your system otherwise. **Valgrind** messages are logged to **stderr**.

## 10.5. SUBSYSTEMS, LOG AND DEBUG SETTINGS

In most cases, you will enable debug logging output by using subsystems and on a temporary basis.

### 10.5.1. Subsystems

Each subsystem has a logging level for its output logs, and for its logs in-memory. You can set different values for each of these subsystems by setting a log file level and a memory level for debug logging. Ceph's logging levels operate on a scale of **1** to **20**, where **1** is terse and **20** is verbose.

A debug logging setting can take a single value for the log level and the memory level, which sets them both as the same value. For example, if you specify **debug_ms = 5**, Ceph will treat it as a log level and a memory level of **5**. You can also specify them separately. The first setting is the log level, and the second setting is the memory level. You must separate them with a forward slash (/). For example, if you want to set the **ms** subsystem's debug logging level to **1** and its memory level to **5**, you would specify it as **debug_ms = 1/5**. For example:

```
debug_<subsystem> = <log-level>/<memory-level>
#for example
debug_osd = 1/20
```

The following table provides a list of Ceph subsystems and their default log and memory levels. Once you complete your logging efforts, restore the subsystems to their default level or to a level suitable for normal operations.

| Subsystem | Log Level | Memory Level |
|---|---|---|
| **default** | 0 | 5 |
| **lockdep** | 0 | 1 |
| **context** | 0 | 1 |

| Subsystem | Log Level | Memory Level |
|---|---|---|
| **`crush`** | 0 | 1 |
| **`buffer`** | 0 | 0 |
| **`timer`** | 0 | 1 |
| **`filer`** | 0 | 1 |
| **`striper`** | 0 | 1 |
| **`objecter`** | 0 | 1 |
| **`rados`** | 0 | 5 |
| **`rbd`** | 0 | 5 |
| **`journaler`** | 0 | 5 |
| **`objectcacher`** | 0 | 5 |
| **`client`** | 0 | 5 |
| **`osd`** | 0 | 5 |
| **`optracker`** | 0 | 5 |
| **`objclass`** | 0 | 5 |
| **`filestore`** | 1 | 3 |

| Subsystem | Log Level | Memory Level |
|---|---|---|
| `journal` | 1 | 3 |
| `ms` | 0 | 5 |
| `mon` | 1 | 5 |
| `monc` | 0 | 10 |
| `paxos` | 1 | 5 |
| `tp` | 0 | 5 |
| `auth` | 1 | 5 |
| `finisher` | 1 | 1 |
| `heartbeatmap` | 1 | 5 |
| `perfcounter` | 1 | 5 |
| `rgw` | 1 | 5 |
| `civetweb` | 1 | 10 |
| `javaclient` | 1 | 5 |
| `asok` | 1 | 5 |
| `throttle` | 1 | 1 |

| Subsystem | Log Level | Memory Level |
|---|---|---|
| **`refs`** | 0 | 0 |
| **`xio`** | 1 | 5 |

Here are examples of the type of messages you will see in the logs when the verbosity is increased for the monitors and the OSDs.

### Monitor Debug Settings

```
debug_ms = 5
debug_mon = 20
debug_paxos = 20
debug_auth = 20
```

### Example Log Output

```
2016-02-12 12:37:04.278761 7f45a9afc700 10 mon.cephn2@0(leader).osd e322
e322: 2 osds: 2 up, 2 in
2016-02-12 12:37:04.278792 7f45a9afc700 10 mon.cephn2@0(leader).osd e322
min_last_epoch_clean 322
2016-02-12 12:37:04.278795 7f45a9afc700 10 mon.cephn2@0(leader).log
v1010106 log
2016-02-12 12:37:04.278799 7f45a9afc700 10 mon.cephn2@0(leader).auth
v2877 auth
2016-02-12 12:37:04.278811 7f45a9afc700 20 mon.cephn2@0(leader) e1
sync_trim_providers
2016-02-12 12:37:09.278914 7f45a9afc700 11 mon.cephn2@0(leader) e1 tick
2016-02-12 12:37:09.278949 7f45a9afc700 10 mon.cephn2@0(leader).pg v8126
v8126: 64 pgs: 64 active+clean; 60168 kB data, 172 MB used, 20285 MB /
20457 MB avail
2016-02-12 12:37:09.278975 7f45a9afc700 10
mon.cephn2@0(leader).paxosservice(pgmap 7511..8126) maybe_trim trim_to
7626 would only trim 115 < paxos_service_trim_min 250
2016-02-12 12:37:09.278982 7f45a9afc700 10 mon.cephn2@0(leader).osd e322
e322: 2 osds: 2 up, 2 in
2016-02-12 12:37:09.278989 7f45a9afc700  5
mon.cephn2@0(leader).paxos(paxos active c 1028850..1029466) is_readable =
1 - now=2016-02-12 12:37:09.278990 lease_expire=0.000000 has v0 lc
1029466
....
2016-02-12 12:59:18.769963 7f45a92fb700  1 -- 192.168.0.112:6789/0 <==
osd.1 192.168.0.114:6800/2801 5724 ==== pg_stats(0 pgs tid 3045 v 0) v1
==== 124+0+0 (2380105412 0 0) 0x5d96300 con 0x4d5bf40
2016-02-12 12:59:18.770053 7f45a92fb700  1 -- 192.168.0.112:6789/0 -->
192.168.0.114:6800/2801 -- pg_stats_ack(0 pgs tid 3045) v1 -- ?+0
0x550ae00 con 0x4d5bf40
2016-02-12 12:59:32.916397 7f45a9afc700  0
mon.cephn2@0(leader).data_health(1) update_stats avail 53% total 1951 MB,
used 780 MB, avail 1053 MB
....
2016-02-12 13:01:05.256263 7f45a92fb700  1 -- 192.168.0.112:6789/0 -->
192.168.0.113:6800/2410 -- mon_subscribe_ack(300s) v1 -- ?+0 0x4f283c0
con 0x4d5b440
```

### OSD Debug Settings

```
debug_ms = 5
debug_osd = 20
debug_filestore = 20
debug_journal = 20
```

### Example Log Output

```
2016-02-12 11:27:53.869151 7f5d55d84700  1 -- 192.168.17.3:0/2410 -->
192.168.17.4:6801/2801 -- osd_ping(ping e322 stamp 2016-02-12
11:27:53.869147) v2 -- ?+0 0x63baa00 con 0x578dee0
2016-02-12 11:27:53.869214 7f5d55d84700  1 -- 192.168.17.3:0/2410 -->
192.168.0.114:6801/2801 -- osd_ping(ping e322 stamp 2016-02-12
11:27:53.869147) v2 -- ?+0 0x638f200 con 0x578e040
2016-02-12 11:27:53.870215 7f5d6359f700  1 -- 192.168.17.3:0/2410 <==
osd.1 192.168.0.114:6801/2801 109210 ==== osd_ping(ping_reply e322 stamp
2016-02-12 11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x63c1a00 con
0x578e040
2016-02-12 11:27:53.870698 7f5d6359f700  1 -- 192.168.17.3:0/2410 <==
osd.1 192.168.17.4:6801/2801 109210 ==== osd_ping(ping_reply e322 stamp
2016-02-12 11:27:53.869147) v2 ==== 47+0+0 (261193640 0 0) 0x6313200 con
0x578dee0
....
2016-02-12 11:28:10.432313 7f5d6e71f700  5 osd.0 322 tick
2016-02-12 11:28:10.432375 7f5d6e71f700 20 osd.0 322 scrub_random_backoff
lost coin flip, randomly backing off
2016-02-12 11:28:10.432381 7f5d6e71f700 10 osd.0 322 do_waiters -- start
2016-02-12 11:28:10.432383 7f5d6e71f700 10 osd.0 322 do_waiters -- finish
```

## 10.5.2. Logging Settings

Logging and debugging settings are not required in a Ceph configuration file, but you can override default settings as needed. Ceph supports the following settings:

**log_file**

**Description**

The location of the logging file for your cluster.

**Type**

String

**Required**

No

**Default**

`/var/log/ceph/$cluster-$name.log`

**log_max_new**

**Description**

The maximum number of new log files.

**Type**

Integer

**Required**

No

**Default**

`1000`

**log_max_recent**

**Description**

The maximum number of recent events to include in a log file.

**Type**

Integer

**Required**

No

**Default**

`1000000`

**log_to_stderr**

**Description**

Determines if logging messages appear in `stderr`.

**Type**

Boolean

**Required**

No

**Default**

`true`

**err_to_stderr**

**Description**

Determines if error messages appear in `stderr`.

**Type**

Boolean

**Required**

No

**Default**

`true`

**log_to_syslog**

**Description**

Determines if logging messages appear in `syslog`.

**Type**

Boolean

**Required**

No

**Default**

`false`

**err_to_syslog**

**Description**

Determines if error messages appear in `syslog`.

**Type**

Boolean

**Required**

No

**Default**

`false`

**log_flush_on_exit**

**Description**

Determines if Ceph flushes the log files after exit.

**Type**

Boolean

**Required**

No

**Default**

`true`

**clog_to_monitors**

**Description**

Determines if `clog` messages will be sent to monitors.

**Type**

Boolean

**Required**

No

**Default**

`true`

**clog_to_syslog**

**Description**

Determines if `clog` messages will be sent to `syslog`.

**Type**

Boolean

**Required**

No

**Default**

`false`

**mon_cluster_log_to_syslog**

**Description**

Determines if the cluster log will be output to `syslog`.

**Type**

Boolean

**Required**

No

**Default**

`false`

**mon_cluster_log_file**

**Description**

The location of the cluster’s log file.

**Type**

String

**Required**

No

**Default**

`/var/log/ceph/$cluster.log`

### 10.5.2.1. OSD

**osd_preserve_trimmed_log**

**Description**

Preserves trimmed logs after trimming.

**Type**

Boolean

**Required**

No

**Default**

`false`

**osd_tmapput_sets_uses_tmap**

**Description**

Uses `tmap`. For debug only.

**Type**

Boolean

**Required**

No

**Default**

`false`

**osd_min_pg_log_entries**

**Description**

The minimum number of log entries for placement groups.

**Type**

32-bit Unsigned Integer

**Required**

No

**Default**

1000

**osd_op_log_threshold**

**Description**

How many op log messages to show up in one pass.

**Type**

Integer

**Required**

No

**Default**

5

### 10.5.2.2. File Store

**filestore_debug_omap_check**

**Description**

Debugging check on synchronization. This is an expensive operation.

**Type**

Boolean

**Required**

No

**Default**

0

### 10.5.2.3. RADOS Gateway

**rgw_log_nonexistent_bucket**

**Description**

Log non-existent buckets.

**Type**

Boolean

**Required**

No

**Default**

`false`

**rgw_log_object_name**

**Description**

Log an object’s name.

**Type**

String

**Required**

No

**Default**

`%Y-%m-%d-%H-%i-%n`

**rgw_log_object_name_utc**

**Description**

Object log name contains UTC.

**Type**

Boolean

**Required**

No

**Default**

`false`

**rgw_enable_ops_log**

**Description**

Enables logging of every RGW operation.

**Type**

Boolean

**Required**

No

**Default**

`true`

**rgw_enable_usage_log**

**Description**

Enable logging of RGW’s bandwidth usage.

**Type**

Boolean

**Required**

No

**Default**

`true`

**rgw_usage_log_flush_threshold**

**Description**

Threshold to flush pending log data.

**Type**

Integer

**Required**

No

**Default**

`1024`

**rgw_usage_log_tick_interval**

**Description**

Flush pending log data every **s** seconds.

**Type**

Integer

**Required**

No

**Default**

30

**rgw_intent_log_object_name**

**Description, Type**

String

**Required**

No

**Default**

**`%Y-%m-%d-%i-%n`**

### rgw_intent_log_object_name utc

**Description**

Include a UTC time stamp in the intent log object name.

**Type**

Boolean

**Required**

No

**Default**

**`false`**