



SUSE Enterprise Storage 7

# Deployment Guide

# Deployment Guide

SUSE Enterprise Storage 7

by Tomáš Bažant, Alexandra Settle, and Liam Proven

Publication Date: 12 Dec 2024

<https://documentation.suse.com> 

Copyright © 2020–2024 SUSE LLC and contributors. All rights reserved.

Except where otherwise noted, this document is licensed under Creative Commons Attribution-ShareAlike 4.0 International (CC-BY-SA 4.0): <https://creativecommons.org/licenses/by-sa/4.0/legalcode> .

For SUSE trademarks, see <http://www.suse.com/company/legal/> . All third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (\*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors nor the translators shall be held liable for possible errors or the consequences thereof.

# Contents

## About this guide **viii**

- 1 Available documentation **viii**
- 2 Giving feedback **ix**
- 3 Documentation conventions **x**
- 4 Support **xii**
  - Support statement for SUSE Enterprise Storage **xii** • Technology previews **xiii**
- 5 Ceph contributors **xiii**
- 6 Commands and command prompts used in this guide **xiv**
  - Salt-related commands **xiv** • Ceph related commands **xiv** • General Linux commands **xv** • Additional information **xvi**

## I INTRODUCING SUSE ENTERPRISE STORAGE (SES) **1**

### 1 SES and Ceph **2**

- 1.1 Ceph features **2**
- 1.2 Ceph core components **3**
  - RADOS **3** • CRUSH **4** • Ceph nodes and daemons **5**
- 1.3 Ceph storage structure **6**
  - Pools **6** • Placement groups **7** • Example **7**
- 1.4 BlueStore **8**
- 1.5 Additional information **10**

### 2 Hardware requirements and recommendations **11**

- 2.1 Network overview **11**
  - Network recommendations **12**

2.2	Multiple architecture configurations	14
2.3	Hardware configuration	15
	Minimum cluster configuration	15
	Recommended production cluster configuration	17
	Multipath configuration	17
2.4	Object Storage Nodes	18
	Minimum requirements	18
	Minimum disk size	19
	Recommended size for the BlueStore's WAL and DB device	19
	SSD for WAL/DB partitions	20
	Maximum recommended number of disks	21
2.5	Monitor nodes	21
2.6	Object Gateway nodes	22
2.7	Metadata Server nodes	22
2.8	Admin Node	22
2.9	iSCSI Gateway nodes	22
2.10	SES and other SUSE products	22
	SUSE Manager	22
2.11	Name limitations	23
2.12	OSD and monitor sharing one server	23
<b>3</b>	<b>Admin Node HA setup</b>	<b>24</b>
3.1	Outline of the HA cluster for Admin Node	24
3.2	Building an HA cluster with the Admin Node	25
<b>II</b>	<b>DEPLOYING CEPH CLUSTER</b>	<b>27</b>
<b>4</b>	<b>Introduction and common tasks</b>	<b>28</b>
4.1	Read the release notes	28

- 5 Installing and configuring SUSE Linux Enterprise Server 30
- 6 Deploying Salt 31
- 7 Deploying the bootstrap cluster using ceph-salt 34
  - 7.1 Installing ceph-salt 34
  - 7.2 Configuring cluster properties 34
    - Using the ceph-salt shell 35
    - Adding Salt Minions 37
    - Specifying Salt Minions managed by cephadm 38
    - Specifying Admin Node 38
    - Specifying first MON/MGR node 38
    - Specifying tuned profiles 39
    - Generating an SSH key pair 39
    - Configuring the time server 40
    - Configuring the Ceph Dashboard login credentials 41
    - Using the container registry 41
    - Enabling data in-flight encryption (msgr2) 46
    - Configuring the cluster network 49
    - Verifying the cluster configuration 49
    - Exporting cluster configurations 51
  - 7.3 Updating nodes and bootstrap minimal cluster 51
  - 7.4 Reviewing final steps 52
  - 7.5 Disable insecure clients 54
- 8 Deploying the remaining core services using cephadm 55
  - 8.1 The **ceph orch** command 55
    - Displaying the orchestrator status 55
    - Listing devices, services, and daemons 55
  - 8.2 Service and placement specification 56
    - Creating service specifications 56
    - Creating placement specification 58
    - Applying cluster specification 58
    - Exporting the specification of a running cluster 59

- 8.3 Deploy Ceph services 59
  - Deploying Ceph Monitors and Ceph Managers 60
  - Deploying Ceph OSDs 61
  - Deploying Metadata Servers 62
  - Deploying Object Gateways 62
  - Deploying iSCSI Gateways 65
  - Deploying NFS Ganesha 67
  - Deploying rbd-mirror 68
  - Deploying the monitoring stack 68

## 9 Deployment of additional services 71

- 9.1 Installation of iSCSI gateway 71
  - iSCSI block storage 71
  - General information about ceph-iscsi 73
  - Deployment considerations 74
  - Installation and configuration 75
  - Exporting RADOS Block Device images using tcmu-runner 82

### III UPGRADING FROM PREVIOUS RELEASES 84

## 10 Upgrade from a previous release 85

- A Ceph maintenance updates based on upstream 'Octopus' point releases 86

## Glossary 91

# About this guide

This guide focuses on deploying a basic Ceph cluster, and how to deploy additional services. It also covers the steps for upgrading to SUSE Enterprise Storage 7 from the previous product version.

SUSE Enterprise Storage 7 is an extension to SUSE Linux Enterprise Server 15 SP2. It combines the capabilities of the Ceph (<http://ceph.com/>) storage project with the enterprise engineering and support of SUSE. SUSE Enterprise Storage 7 provides IT organizations with the ability to deploy a distributed storage architecture that can support a number of use cases using commodity hardware platforms.

## 1 Available documentation



### Note: Online documentation and latest updates

Documentation for our products is available at <https://documentation.suse.com>, where you can also find the latest updates, and browse or download the documentation in various formats. The latest documentation updates can be found in the English language version.

In addition, the product documentation is available in your installed system under `/usr/share/doc/manual`. It is included in an RPM package named `ses-manual_LANG_CODE`. Install it if it is not already on your system, for example:

```
# zypper install ses-manual_en
```

The following documentation is available for this product:

*Deployment Guide* (<https://documentation.suse.com/ses/html/ses-all/book-storage-deployment.html>)

This guide focuses on deploying a basic Ceph cluster, and how to deploy additional services. It also cover the steps for upgrading to SUSE Enterprise Storage 7 from the previous product version.

*Administration and Operations Guide* (<https://documentation.suse.com/ses/html/ses-all/book-storage-admin.html>) ↗

This guide focuses on routine tasks that you as an administrator need to take care of after the basic Ceph cluster has been deployed (day 2 operations). It also describes all the supported ways to access data stored in a Ceph cluster.

*Security Hardening Guide* (<https://documentation.suse.com/ses/html/ses-all/book-storage-security.html>) ↗

This guide focuses on how to ensure your cluster is secure.

*Troubleshooting Guide* (<https://documentation.suse.com/ses/html/ses-all/book-storage-troubleshooting.html>) ↗

This guide takes you through various common problems when running SUSE Enterprise Storage 7 and other related issues to relevant components such as Ceph or Object Gateway.

*SUSE Enterprise Storage for Windows Guide* (<https://documentation.suse.com/ses/html/ses-all/book-storage-windows.html>) ↗

This guide describes the integration, installation, and configuration of Microsoft Windows environments and SUSE Enterprise Storage using the Windows Driver.

## 2 Giving feedback

We welcome feedback on, and contributions to, this documentation. There are several channels for this:

### Service requests and support

For services and support options available for your product, see <http://www.suse.com/support/> ↗.

To open a service request, you need a SUSE subscription registered at SUSE Customer Center. Go to <https://scc.suse.com/support/requests> ↗, log in, and click *Create New*.

### Bug reports

Report issues with the documentation at <https://bugzilla.suse.com/> ↗. Reporting issues requires a Bugzilla account.

To simplify this process, you can use the *Report Documentation Bug* links next to headlines in the HTML version of this document. These preselect the right product and category in Bugzilla and add a link to the current section. You can start typing your bug report right away.

## Contributions

To contribute to this documentation, use the *Edit Source* links next to headlines in the HTML version of this document. They take you to the source code on GitHub, where you can open a pull request. Contributing requires a GitHub account.

For more information about the documentation environment used for this documentation, see the repository's README at <https://github.com/SUSE/doc-ses>.

## Mail

You can also report errors and send feedback concerning the documentation to [doc-team@suse.com](mailto:doc-team@suse.com). Include the document title, the product version, and the publication date of the document. Additionally, include the relevant section number and title (or provide the URL) and provide a concise description of the problem.

# 3 Documentation conventions

The following notices and typographic conventions are used in this document:

- /etc/passwd: Directory names and file names
- PLACEHOLDER: Replace PLACEHOLDER with the actual value
- PATH: An environment variable
- ls, --help: Commands, options, and parameters
- user: The name of user or group
- package\_name: The name of a software package
- **Alt**, **Alt - F1**: A key to press or a key combination. Keys are shown in uppercase as on a keyboard.
- *File*, *File > Save As*: menu items, buttons
- **AMD/Intel** This paragraph is only relevant for the AMD64/Intel 64 architectures. The arrows mark the beginning and the end of the text block. ◁
- **IBM Z, POWER** This paragraph is only relevant for the architectures IBM Z and POWER. The arrows mark the beginning and the end of the text block. ◁
- *Chapter 1, “Example chapter”*: A cross-reference to another chapter in this guide.

- Commands that must be run with `root` privileges. Often you can also prefix these commands with the `sudo` command to run them as non-privileged user.

```
# command  
> sudo command
```

- Commands that can be run by non-privileged users.

```
> command
```

- Notices



### Warning: Warning notice

Vital information you must be aware of before proceeding. Warns you about security issues, potential loss of data, damage to hardware, or physical hazards.



### Important: Important notice

Important information you should be aware of before proceeding.



### Note: Note notice

Additional information, for example about differences in software versions.



### Tip: Tip notice

Helpful information, like a guideline or a piece of practical advice.

- Compact Notices



Additional information, for example about differences in software versions.



Helpful information, like a guideline or a piece of practical advice.

## 4 Support

Find the support statement for SUSE Enterprise Storage and general information about technology previews below. For details about the product lifecycle, see <https://www.suse.com/lifecycle>. If you are entitled to support, find details on how to collect information for a support ticket at <https://documentation.suse.com/sles-15/html/SLES-all/cha-adm-support.html>.

### 4.1 Support statement for SUSE Enterprise Storage

To receive support, you need an appropriate subscription with SUSE. To view the specific support offerings available to you, go to <https://www.suse.com/support/> and select your product.

The support levels are defined as follows:

#### L1

Problem determination, which means technical support designed to provide compatibility information, usage support, ongoing maintenance, information gathering and basic troubleshooting using available documentation.

#### L2

Problem isolation, which means technical support designed to analyze data, reproduce customer problems, isolate problem area and provide a resolution for problems not resolved by Level 1 or prepare for Level 3.

#### L3

Problem resolution, which means technical support designed to resolve problems by engaging engineering to resolve product defects which have been identified by Level 2 Support.

For contracted customers and partners, SUSE Enterprise Storage is delivered with L3 support for all packages, except for the following:

- Technology previews.
- Sound, graphics, fonts, and artwork.
- Packages that require an additional customer contract.
- Some packages shipped as part of the module *Workstation Extension* are L2-supported only.
- Packages with names ending in `-devel` (containing header files and similar developer resources) will only be supported together with their main packages.

SUSE will only support the usage of original packages. That is, packages that are unchanged and not recompiled.

## 4.2 Technology previews

Technology previews are packages, stacks, or features delivered by SUSE to provide glimpses into upcoming innovations. Technology previews are included for your convenience to give you a chance to test new technologies within your environment. We would appreciate your feedback! If you test a technology preview, please contact your SUSE representative and let them know about your experience and use cases. Your input is helpful for future development.

Technology previews have the following limitations:

- Technology previews are still in development. Therefore, they may be functionally incomplete, unstable, or in other ways *not* suitable for production use.
- Technology previews are *not* supported.
- Technology previews may only be available for specific hardware architectures.
- Details and functionality of technology previews are subject to change. As a result, upgrading to subsequent releases of a technology preview may be impossible and require a fresh installation.
- SUSE may discover that a preview does not meet customer or market needs, or does not comply with enterprise standards. Technology previews can be removed from a product at any time. SUSE does not commit to providing a supported version of such technologies in the future.

For an overview of technology previews shipped with your product, see the release notes at [https://www.suse.com/releasenotes/x86\\_64/SUSE-Enterprise-Storage/7](https://www.suse.com/releasenotes/x86_64/SUSE-Enterprise-Storage/7).

## 5 Ceph contributors

The Ceph project and its documentation is a result of the work of hundreds of contributors and organizations. See <https://ceph.com/contributors/> for more details.

## 6 Commands and command prompts used in this guide

As a Ceph cluster administrator, you will be configuring and adjusting the cluster behavior by running specific commands. There are several types of commands you will need:

### 6.1 Salt-related commands

These commands help you to deploy Ceph cluster nodes, run commands on several (or all) cluster nodes at the same time, or assist you when adding or removing cluster nodes. The most frequently used commands are `ceph-salt` and `ceph-salt config`. You need to run Salt commands on the Salt Master node as `root`. These commands are introduced with the following prompt:

```
root@master #
```

For example:

```
root@master # ceph-salt config ls
```

### 6.2 Ceph related commands

These are lower-level commands to configure and fine tune all aspects of the cluster and its gateways on the command line, for example `ceph`, `cephadm`, `rbd`, or `radosgw-admin`.

To run Ceph related commands, you need to have read access to a Ceph key. The key's capabilities then define your privileges within the Ceph environment. One option is to run Ceph commands as `root` (or via `sudo`) and use the unrestricted default keyring 'ceph.client.admin.key'. The safer and recommended option is to create a more restrictive individual key for each administrator user and put it in a directory where the users can read it, for example:

```
~/ceph/ceph.client.USERNAME.keyring
```



#### Tip: Path to Ceph keys

To use a custom admin user and keyring, you need to specify the user name and path to the key each time you run the `ceph` command using the `-n client.USER_NAME` and `--keyring PATH/TO/KEYRING` options.

To avoid this, include these options in the `CEPH_ARGS` variable in the individual users' `~/.bashrc` files.

Although you can run Ceph-related commands on any cluster node, we recommend running them on the Admin Node. This documentation uses the `cephuser` user to run the commands, therefore they are introduced with the following prompt:

```
cephuser@adm >
```

For example:

```
cephuser@adm > ceph auth list
```



### Tip: Commands for specific nodes

If the documentation instructs you to run a command on a cluster node with a specific role, it will be addressed by the prompt. For example:

```
cephuser@mon >
```

## 6.2.1 Running `ceph-volume`

Starting with SUSE Enterprise Storage 7, Ceph services are running containerized. If you need to run `ceph-volume` on an OSD node, you need to prepend it with the `cephadm` command, for example:

```
cephuser@adm > cephadm ceph-volume simple scan
```

## 6.3 General Linux commands

Linux commands not related to Ceph, such as `mount`, `cat`, or `openssl`, are introduced either with the `cephuser@adm >` or `#` prompts, depending on which privileges the related command requires.

## 6.4 Additional information

For more information on Ceph key management, refer to *Book "Administration and Operations Guide", Chapter 30 "Authentication with cephx", Section 30.2 "Key management"*.

# I Introducing SUSE Enterprise Storage (SES)

- 1 SES and Ceph 2
- 2 Hardware requirements and recommendations 11
- 3 Admin Node HA setup 24

# 1 SES and Ceph

SUSE Enterprise Storage is a distributed storage system designed for scalability, reliability and performance which is based on the Ceph technology. A Ceph cluster can be run on commodity servers in a common network like Ethernet. The cluster scales up well to thousands of servers (later on referred to as nodes) and into the petabyte range. As opposed to conventional systems which have allocation tables to store and fetch data, Ceph uses a deterministic algorithm to allocate storage for data and has no centralized information structure. Ceph assumes that in storage clusters the addition or removal of hardware is the rule, not the exception. The Ceph cluster automates management tasks such as data distribution and redistribution, data replication, failure detection and recovery. Ceph is both self-healing and self-managing which results in a reduction of administrative and budget overhead.

This chapter provides a high level overview of SUSE Enterprise Storage 7 and briefly describes the most important components.

## 1.1 Ceph features

The Ceph environment has the following features:

### Scalability

Ceph can scale to thousands of nodes and manage storage in the range of petabytes.

### Commodity Hardware

No special hardware is required to run a Ceph cluster. For details, see [Chapter 2, Hardware requirements and recommendations](#)

### Self-managing

The Ceph cluster is self-managing. When nodes are added, removed or fail, the cluster automatically redistributes the data. It is also aware of overloaded disks.

### No Single Point of Failure

No node in a cluster stores important information alone. The number of redundancies can be configured.

### Open Source Software

Ceph is an open source software solution and independent of specific hardware or vendors.

## 1.2 Ceph core components

To make full use of Ceph's power, it is necessary to understand some of the basic components and concepts. This section introduces some parts of Ceph that are often referenced in other chapters.

### 1.2.1 RADOS

The basic component of Ceph is called *RADOS (Reliable Autonomic Distributed Object Store)*. It is responsible for managing the data stored in the cluster. Data in Ceph is usually stored as objects. Each object consists of an identifier and the data.

RADOS provides the following access methods to the stored objects that cover many use cases:

#### Object Gateway

Object Gateway is an HTTP REST gateway for the RADOS object store. It enables direct access to objects stored in the Ceph cluster.

#### RADOS Block Device

RADOS Block Device (RBD) can be accessed like any other block device. These can be used for example in combination with libvirt for virtualization purposes.

#### CephFS

The Ceph File System is a POSIX-compliant file system.

#### librados

librados is a library that can be used with many programming languages to create an application capable of directly interacting with the storage cluster.

librados is used by Object Gateway and RBD while CephFS directly interfaces with RADOS *Figure 1.1, "Interfaces to the Ceph object store".*

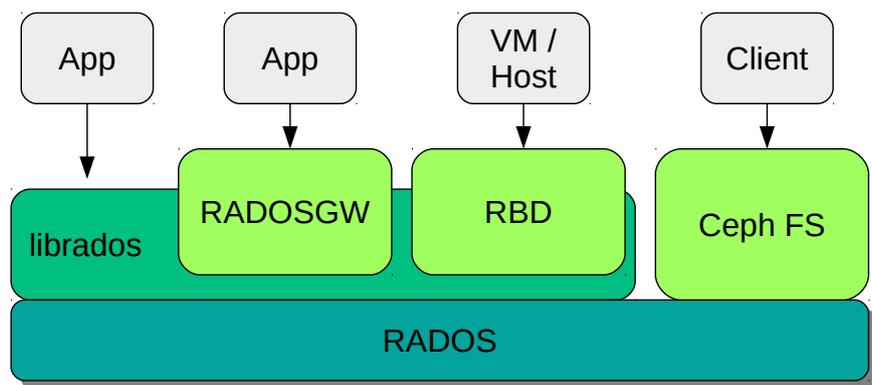


FIGURE 1.1: INTERFACES TO THE CEPH OBJECT STORE

## 1.2.2 CRUSH

At the core of a Ceph cluster is the *CRUSH* algorithm. CRUSH is the acronym for *Controlled Replication Under Scalable Hashing*. CRUSH is a function that handles the storage allocation and needs comparably few parameters. That means only a small amount of information is necessary to calculate the storage position of an object. The parameters are a current map of the cluster including the health state, some administrator-defined placement rules and the name of the object that needs to be stored or retrieved. With this information, all nodes in the Ceph cluster are able to calculate where an object and its replicas are stored. This makes writing or reading data very efficient. CRUSH tries to evenly distribute data over all nodes in the cluster.

The *CRUSH Map* contains all storage nodes and administrator-defined placement rules for storing objects in the cluster. It defines a hierarchical structure that usually corresponds to the physical structure of the cluster. For example, the data-containing disks are in hosts, hosts are in racks, racks in rows and rows in data centers. This structure can be used to define *failure domains*. Ceph then ensures that replications are stored on different branches of a specific failure domain.

If the failure domain is set to rack, replications of objects are distributed over different racks. This can mitigate outages caused by a failed switch in a rack. If one power distribution unit supplies a row of racks, the failure domain can be set to row. When the power distribution unit fails, the replicated data is still available on other rows.

### 1.2.3 Ceph nodes and daemons

In Ceph, nodes are servers working for the cluster. They can run several different types of daemons. We recommend running only one type of daemon on each node, except for Ceph Manager daemons which can be co-located with Ceph Monitors. Each cluster requires at least Ceph Monitor, Ceph Manager, and Ceph OSD daemons:

#### Admin Node

The *Admin Node* is a Ceph cluster node from which you run commands to manage the cluster. The Admin Node is a central point of the Ceph cluster because it manages the rest of the cluster nodes by querying and instructing their Salt Minion services.

#### Ceph Monitor

*Ceph Monitor* (often abbreviated as *MON*) nodes maintain information about the cluster health state, a map of all nodes and data distribution rules (see [Section 1.2.2, "CRUSH"](#)).

If failures or conflicts occur, the Ceph Monitor nodes in the cluster decide by majority which information is correct. To form a qualified majority, it is recommended to have an odd number of Ceph Monitor nodes, and at least three of them.

If more than one site is used, the Ceph Monitor nodes should be distributed over an odd number of sites. The number of Ceph Monitor nodes per site should be such that more than 50% of the Ceph Monitor nodes remain functional if one site fails.

#### Ceph Manager

The Ceph Manager collects the state information from the whole cluster. The Ceph Manager daemon runs alongside the Ceph Monitor daemons. It provides additional monitoring, and interfaces the external monitoring and management systems. It includes other services as well. For example, the Ceph Dashboard Web UI runs on the same node as the Ceph Manager.

The Ceph Manager requires no additional configuration, beyond ensuring it is running.

#### Ceph OSD

A *Ceph OSD* is a daemon handling *Object Storage Devices* which are a physical or logical storage units (hard disks or partitions). Object Storage Devices can be physical disks/partitions or logical volumes. The daemon additionally takes care of data replication and rebalancing in case of added or removed nodes.

Ceph OSD daemons communicate with monitor daemons and provide them with the state of the other OSD daemons.

To use CephFS, Object Gateway, NFS Ganesha, or iSCSI Gateway, additional nodes are required:

#### Metadata Server (MDS)

CephFS metadata is stored in its own RADOS pool (see [Section 1.3.1, “Pools”](#)). The Metadata Servers act as a smart caching layer for the metadata and serializes access when needed. This allows concurrent access from many clients without explicit synchronization.

#### Object Gateway

The Object Gateway is an HTTP REST gateway for the RADOS object store. It is compatible with OpenStack Swift and Amazon S3 and has its own user management.

#### NFS Ganesha

NFS Ganesha provides an NFS access to either the Object Gateway or the CephFS. It runs in the user instead of the kernel space and directly interacts with the Object Gateway or CephFS.

#### iSCSI Gateway

iSCSI is a storage network protocol that allows clients to send SCSI commands to SCSI storage devices (targets) on remote servers.

#### Samba Gateway

The Samba Gateway provides a Samba access to data stored on CephFS.

## 1.3 Ceph storage structure

### 1.3.1 Pools

Objects that are stored in a Ceph cluster are put into *pools*. Pools represent logical partitions of the cluster to the outside world. For each pool a set of rules can be defined, for example, how many replications of each object must exist. The standard configuration of pools is called *replicated pool*.

Pools usually contain objects but can also be configured to act similar to a RAID 5. In this configuration, objects are stored in chunks along with additional coding chunks. The coding chunks contain the redundant information. The number of data and coding chunks can be defined by the administrator. In this configuration, pools are referred to as *erasure coded pools* or *EC pools*.

## 1.3.2 Placement groups

*Placement Groups* (PGs) are used for the distribution of data within a pool. When creating a pool, a certain number of placement groups is set. The placement groups are used internally to group objects and are an important factor for the performance of a Ceph cluster. The PG for an object is determined by the object's name.

## 1.3.3 Example

This section provides a simplified example of how Ceph manages data (see [Figure 1.2, "Small scale Ceph example"](#)). This example does not represent a recommended configuration for a Ceph cluster. The hardware setup consists of three storage nodes or Ceph OSDs (Host 1, Host 2, Host 3). Each node has three hard disks which are used as OSDs (osd.1 to osd.9). The Ceph Monitor nodes are neglected in this example.



### Note: Difference between Ceph OSD and OSD

While *Ceph OSD* or *Ceph OSD daemon* refers to a daemon that is run on a node, the word *OSD* refers to the logical disk that the daemon interacts with.

The cluster has two pools, Pool A and Pool B. While Pool A replicates objects only two times, resilience for Pool B is more important and it has three replications for each object.

When an application puts an object into a pool, for example via the REST API, a Placement Group (PG1 to PG4) is selected based on the pool and the object name. The CRUSH algorithm then calculates on which OSDs the object is stored, based on the Placement Group that contains the object.

In this example the failure domain is set to host. This ensures that replications of objects are stored on different hosts. Depending on the replication level set for a pool, the object is stored on two or three OSDs that are used by the Placement Group.

An application that writes an object only interacts with one Ceph OSD, the primary Ceph OSD. The primary Ceph OSD takes care of replication and confirms the completion of the write process after all other OSDs have stored the object.

If osd.5 fails, all object in PG1 are still available on osd.1. As soon as the cluster recognizes that an OSD has failed, another OSD takes over. In this example osd.4 is used as a replacement for osd.5. The objects stored on osd.1 are then replicated to osd.4 to restore the replication level.

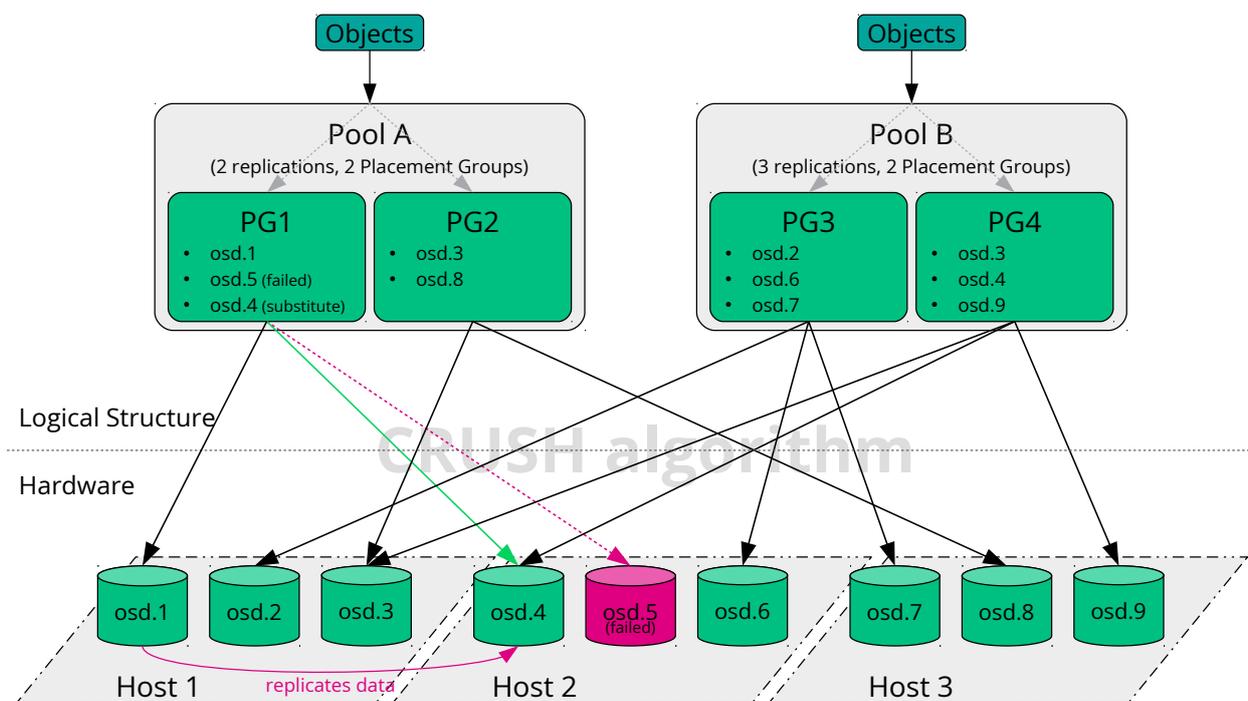


FIGURE 1.2: SMALL SCALE CEPH EXAMPLE

If a new node with new OSDs is added to the cluster, the cluster map is going to change. The CRUSH function then returns different locations for objects. Objects that receive new locations will be relocated. This process results in a balanced usage of all OSDs.

## 1.4 BlueStore

BlueStore is a new default storage back-end for Ceph from SES 5. It has better performance than FileStore, full data check-summing, and built-in compression.

BlueStore manages either one, two, or three storage devices. In the simplest case, BlueStore consumes a single primary storage device. The storage device is normally partitioned into two parts:

1. A small partition named BlueFS that implements file system-like functionalities required by RocksDB.
2. The rest of the device is normally a large partition occupying the rest of the device. It is managed directly by BlueStore and contains all of the actual data. This primary device is normally identified by a block symbolic link in the data directory.

It is also possible to deploy BlueStore across two additional devices:

A *WAL device* can be used for BlueStore's internal journal or write-ahead log. It is identified by the `block.wal` symbolic link in the data directory. It is only useful to use a separate WAL device if the device is faster than the primary device or the DB device, for example when:

- The WAL device is an NVMe, and the DB device is an SSD, and the data device is either SSD or HDD.
- Both the WAL and DB devices are separate SSDs, and the data device is an SSD or HDD.

A *DB device* can be used for storing BlueStore's internal metadata. BlueStore (or rather, the embedded RocksDB) will put as much metadata as it can on the DB device to improve performance. Again, it is only helpful to provision a shared DB device if it is faster than the primary device.



### Tip: Plan for the DB size

Plan thoroughly to ensure sufficient size of the DB device. If the DB device fills up, metadata will spill over to the primary device, which badly degrades the OSD's performance. You can check if a WAL/DB partition is getting full and spilling over with the `ceph daemon osd.ID perf dump` command. The `slow_used_bytes` value shows the amount of data being spilled out:

```
cephuser@adm > ceph daemon osd.ID perf dump | jq '.bluefs'
"db_total_bytes": 1073741824,
"db_used_bytes": 33554432,
"wal_total_bytes": 0,
"wal_used_bytes": 0,
"slow_total_bytes": 554432,
"slow_used_bytes": 554432,
```

## 1.5 Additional information

- Ceph as a community project has its own extensive online documentation. For topics not found in this manual, refer to <https://docs.ceph.com/en/octopus/>.
- The original publication *CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data* by S.A. Weil, S.A. Brandt, E.L. Miller, C. Maltzahn provides helpful insight into the inner workings of Ceph. Especially when deploying large scale clusters it is a recommended reading. The publication can be found at <http://www.ssrc.ucsc.edu/papers/weil-sc06.pdf>.
- SUSE Enterprise Storage can be used with non-SUSE OpenStack distributions. The Ceph clients need to be at a level that is compatible with SUSE Enterprise Storage.



### Note

SUSE supports the server component of the Ceph deployment and the client is supported by the OpenStack distribution vendor.

## 2 Hardware requirements and recommendations

The hardware requirements of Ceph are heavily dependent on the IO workload. The following hardware requirements and recommendations should be considered as a starting point for detailed planning.

In general, the recommendations given in this section are on a per-process basis. If several processes are located on the same machine, the CPU, RAM, disk and network requirements need to be added up.

### 2.1 Network overview

Ceph has several logical networks:

- A front-end network called the public network.
- A trusted internal network, the back-end network, called the cluster network. This is optional.
- One or more client networks for gateways. This is optional and beyond the scope of this chapter.

The public network is the network over which Ceph daemons communicate with each other and with their clients. This means that all Ceph cluster traffic goes over this network except in the case when a cluster network is configured.

The cluster network is the back-end network between the OSD nodes, for replication, re-balancing, and recovery. If configured, this optional network would ideally provide twice the bandwidth of the public network with default three-way replication, since the primary OSD sends two copies to other OSDs via this network. The public network is between clients and gateways on the one side to talk to monitors, managers, MDS nodes, OSD nodes. It is also used by monitors, managers, and MDS nodes to talk with OSD nodes.

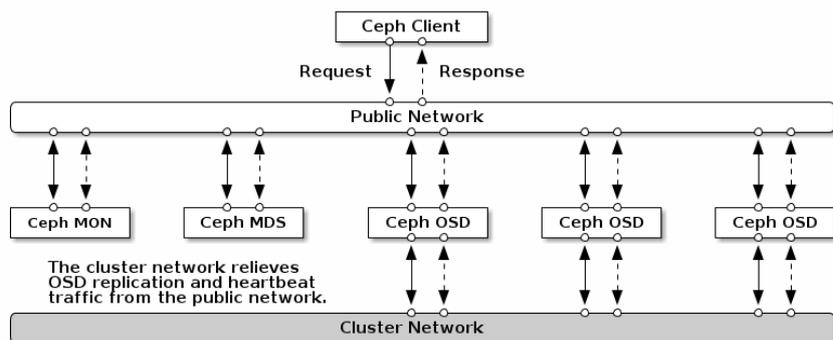


FIGURE 2.1: NETWORK OVERVIEW

### 2.1.1 Network recommendations

We recommend a single fault-tolerant network with enough bandwidth to fulfil your requirements. For the Ceph public network environment, we recommend two bonded 25 GbE (or faster) network interfaces bonded using 802.3ad (LACP). This is considered the minimal setup for Ceph. If you are also using a cluster network, we recommend four bonded 25 GbE network interfaces. Bonding two or more network interfaces provides better throughput via link aggregation and, given redundant links and switches, improved fault tolerance and maintainability.

You can also create VLANs to isolate different types of traffic over a bond. For example, you can create a bond to provide two VLAN interfaces, one for the public network, and the second for the cluster network. However, this is *not* required when setting up Ceph networking. Details on bonding the interfaces can be found in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-network.html#sec-network-iface-bonding>.

Fault tolerance can be enhanced through isolating the components into failure domains. To improve fault tolerance of the network, bonding one interface from two separate Network Interface Cards (NIC) offers protection against failure of a single NIC. Similarly, creating a bond across two switches protects against failure of a switch. We recommend consulting with the network equipment vendor in order to architect the level of fault tolerance required.

#### ! Important: Administration network not supported

Additional administration network setup—that enables for example separating SSH, Salt, or DNS networking—is neither tested nor supported.



## Tip: Nodes configured via DHCP

If your storage nodes are configured via DHCP, the default timeouts may not be sufficient for the network to be configured correctly before the various Ceph daemons start. If this happens, the Ceph MONs and OSDs will not start correctly (running `systemctl status ceph\*` will result in "unable to bind" errors). To avoid this issue, we recommend increasing the DHCP client timeout to at least 30 seconds on each node in your storage cluster. This can be done by changing the following settings on each node:

In `/etc/sysconfig/network/dhcp`, set

```
DHCLIENT_WAIT_AT_BOOT="30"
```

In `/etc/sysconfig/network/config`, set

```
WAIT_FOR_INTERFACES="60"
```

### 2.1.1.1 Adding a private network to a running cluster

If you do not specify a cluster network during Ceph deployment, it assumes a single public network environment. While Ceph operates fine with a public network, its performance and security improves when you set a second private cluster network. To support two networks, each Ceph node needs to have at least two network cards.

You need to apply the following changes to each Ceph node. It is relatively quick to do for a small cluster, but can be very time consuming if you have a cluster consisting of hundreds or thousands of nodes.

1. Set the cluster network using the following command:

```
# ceph config set global cluster_network MY_NETWORK
```

Restart the OSDs to bind to the specified cluster network:

```
# systemctl restart ceph-*@osd.*.service
```

2. Check that the private cluster network works as expected on the OS level.

### 2.1.1.2 Monitoring nodes on different subnets

If the monitor nodes are on multiple subnets, for example they are located in different rooms and served by different switches, you need to specify their public network address in CIDR notation:

```
cephuser@adm > ceph config set mon public_network  
"MON_NETWORK_1, MON_NETWORK_2, MON_NETWORK_N"
```

For example:

```
cephuser@adm > ceph config set mon public_network "192.168.1.0/24, 10.10.0.0/16"
```



#### Warning

If you do specify more than one network segment for the public (or cluster) network as described in this section, each of these subnets must be capable of routing to all the others - otherwise, the MONs and other Ceph daemons on different network segments will not be able to communicate and a split cluster will ensue. Additionally, if you are using a firewall, make sure you include each IP address or subnet in your iptables and open ports for them on all nodes as necessary.

## 2.2 Multiple architecture configurations

SUSE Enterprise Storage supports both x86 and Arm architectures. When considering each architecture, it is important to note that from a cores per OSD, frequency, and RAM perspective, there is no real difference between CPU architectures for sizing.

As with smaller x86 processors (non-server), lower-performance Arm-based cores may not provide an optimal experience, especially when used for erasure coded pools.



#### Note

Throughout the documentation, *SYSTEM-ARCH* is used in place of x86 or Arm.

## 2.3 Hardware configuration

For the best product experience, we recommend to start with the recommended cluster configuration. For a test cluster or a cluster with less performance requirements, we document a minimal supported cluster configuration.

### 2.3.1 Minimum cluster configuration

A minimal product cluster configuration consists of:

- At least four physical nodes (OSD nodes) with co-location of services
- Dual-10 Gb Ethernet as a bonded network
- A separate Admin Node (can be virtualized on an external node)

A detailed configuration is:

- Separate Admin Node with 4 GB RAM, four cores, 1 TB storage capacity. This is typically the Salt Master node. Ceph services and gateways, such as Ceph Monitor, Metadata Server, Ceph OSD, Object Gateway, or NFS Ganesha are not supported on the Admin Node as it needs to orchestrate the cluster update and upgrade processes independently.
- At least four physical OSD nodes, with eight OSD disks each, see [Section 2.4.1, “Minimum requirements”](#) for requirements.  
The total capacity of the cluster should be sized so that even with one node unavailable, the total used capacity (including redundancy) does not exceed 80%.
- Three Ceph Monitor instances. Monitors need to be run from SSD/NVMe storage, not HDDs, for latency reasons.
- Monitors, Metadata Server, and gateways can be co-located on the OSD nodes, see [Section 2.12, “OSD and monitor sharing one server”](#) for monitor co-location. If you co-locate services, the memory and CPU requirements need to be added up.
- iSCSI Gateway, Object Gateway, and Metadata Server require at least incremental 4 GB RAM and four cores.
- If you are using CephFS, S3/Swift, iSCSI, at least two instances of the respective roles (Metadata Server, Object Gateway, iSCSI) are required for redundancy and availability.

- The nodes are to be dedicated to SUSE Enterprise Storage and must not be used for any other physical, containerized, or virtualized workload.
- If any of the gateways (iSCSI, Object Gateway, NFS Ganesha, Metadata Server, ...) are deployed within VMs, these VMs must not be hosted on the physical machines serving other cluster roles. (This is unnecessary, as they are supported as collocated services.)
- When deploying services as VMs on hypervisors outside the core physical cluster, failure domains must be respected to ensure redundancy.  
For example, do not deploy multiple roles of the same type on the same hypervisor, such as multiple MONs or MDSs instances.
- When deploying inside VMs, it is particularly crucial to ensure that the nodes have strong network connectivity and well working time synchronization.
- The hypervisor nodes must be adequately sized to avoid interference by other workloads consuming CPU, RAM, network, and storage resources.

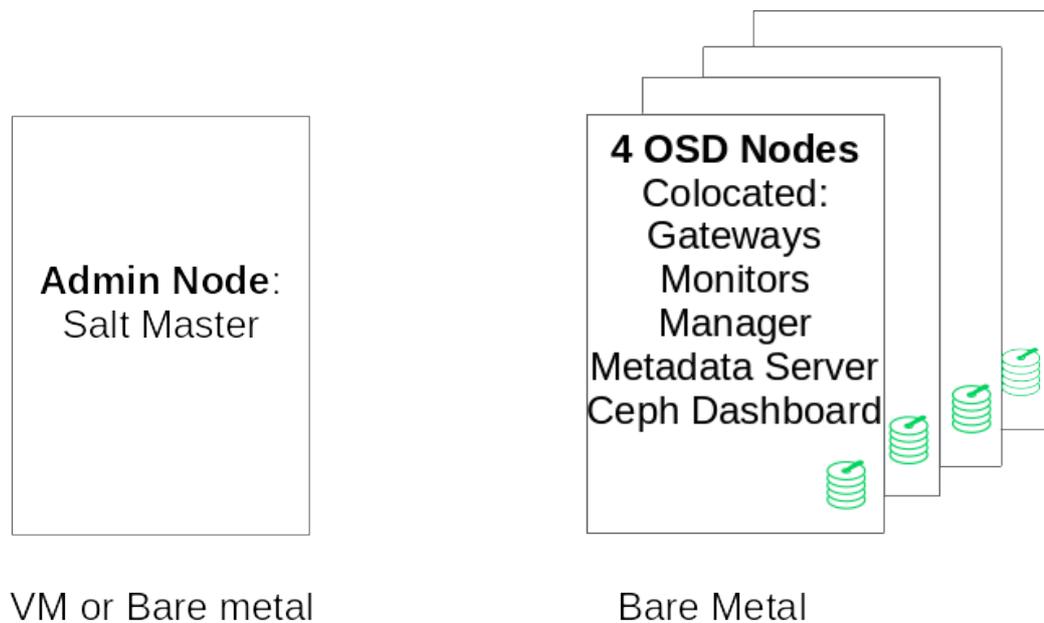


FIGURE 2.2: MINIMUM CLUSTER CONFIGURATION

## 2.3.2 Recommended production cluster configuration

Once you grow your cluster, we recommend relocating Ceph Monitors, Metadata Servers, and Gateways to separate nodes for better fault tolerance.

- Seven Object Storage Nodes
  - No single node exceeds ~15% of total storage.
  - The total capacity of the cluster should be sized so that even with one node unavailable, the total used capacity (including redundancy) does not exceed 80%.
  - 25 Gb Ethernet or better, bonded for internal cluster and external public network each.
  - 56+ OSDs per storage cluster.
  - See [Section 2.4.1, “Minimum requirements”](#) for further recommendation.
- Dedicated physical infrastructure nodes.
  - Three Ceph Monitor nodes: 4 GB RAM, 4 core processor, RAID 1 SSDs for disk. See [Section 2.5, “Monitor nodes”](#) for further recommendation.
  - Object Gateway nodes: 32 GB RAM, 8 core processor, RAID 1 SSDs for disk. See [Section 2.6, “Object Gateway nodes”](#) for further recommendation.
  - iSCSI Gateway nodes: 16 GB RAM, 8 core processor, RAID 1 SSDs for disk. See [Section 2.9, “iSCSI Gateway nodes”](#) for further recommendation.
  - Metadata Server nodes (one active/one hot standby): 32 GB RAM, 8 core processor, RAID 1 SSDs for disk. See [Section 2.7, “Metadata Server nodes”](#) for further recommendation.
  - One SES Admin Node: 4 GB RAM, 4 core processor, RAID 1 SSDs for disk.

## 2.3.3 Multipath configuration

If you want to use multipath hardware, ensure that LVM sees `multipath_component_detection = 1` in the configuration file under the `devices` section. This can be checked via the `lvm config` command.

Alternatively, ensure that LVM filters a device's mpath components via the LVM filter configuration. This will be host specific.



## Note

This is not recommended and should only ever be considered if `multipath_component_detection = 1` cannot be set.

For more information on multipath configuration, see <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-multipath.html#sec-multipath-lvm>.

## 2.4 Object Storage Nodes

### 2.4.1 Minimum requirements

- The following CPU recommendations account for devices independent of usage by Ceph:
  - 1x 2GHz CPU Thread per spinner.
  - 2x 2GHz CPU Thread per SSD.
  - 4x 2GHz CPU Thread per NVMe.
- Separate 10 GbE networks (public/client and internal), required 4x 10 GbE, recommended 2x 25 GbE.
- Total RAM required = number of OSDs x (1 GB + `osd_memory_target`) + 16 GB  
Refer to *Book "Administration and Operations Guide", Chapter 28 "Ceph cluster configuration", Section 28.4.1 "Configuring automatic cache sizing"* for more details on `osd_memory_target`.
- OSD disks in JBOD configurations or individual RAID-0 configurations.
- OSD journal can reside on OSD disk.
- OSD disks should be exclusively used by SUSE Enterprise Storage.
- Dedicated disk and SSD for the operating system, preferably in a RAID 1 configuration.
- Allocate at least an additional 4 GB of RAM if this OSD host will host part of a cache pool used for cache tiering.

- Ceph Monitors, gateway and Metadata Servers can reside on Object Storage Nodes.
- For disk performance reasons, OSD nodes are bare metal nodes. No other workloads should run on an OSD node unless it is a minimal setup of Ceph Monitors and Ceph Managers.
- SSDs for Journal with 6:1 ratio SSD journal to OSD.



## Note

Ensure that OSD nodes do not have any networked block devices mapped, such as iSCSI or RADOS Block Device images.

### 2.4.2 Minimum disk size

There are two types of disk space needed to run on OSD: the space for the WAL/DB device, and the primary space for the stored data. The minimum (and default) value for the WAL/DB is 6 GB. The minimum space for data is 5 GB, as partitions smaller than 5 GB are automatically assigned the weight of 0.

So although the minimum disk space for an OSD is 11 GB, we do not recommend a disk smaller than 20 GB, even for testing purposes.

### 2.4.3 Recommended size for the BlueStore's WAL and DB device



## Tip: More Information

Refer to [Section 1.4, "BlueStore"](#) for more information on BlueStore.

- We recommend reserving 4 GB for the WAL device. While the minimal DB size is 64 GB for RBD-only workloads, the recommended DB size for Object Gateway and CephFS workloads is 2% of the main device capacity (but at least 196 GB).



## Important

We recommend larger DB volumes for high-load deployments, especially if there is high RGW or CephFS usage. Reserve some capacity (slots) to install more hardware for more DB space if required.

- If you intend to put the WAL and DB device on the same disk, then we recommend using a single partition for both devices, rather than having a separate partition for each. This allows Ceph to use the DB device for the WAL operation as well. Management of the disk space is therefore more effective as Ceph uses the DB partition for the WAL only if there is a need for it. Another advantage is that the probability that the WAL partition gets full is very small, and when it is not used fully then its space is not wasted but used for DB operation.

To share the DB device with the WAL, do *not* specify the WAL device, and specify only the DB device.

Find more information about specifying an OSD layout in *Book "Administration and Operations Guide", Chapter 13 "Operational tasks", Section 13.4.3 "Adding OSDs using DriveGroups specification"*.

## 2.4.4 SSD for WAL/DB partitions

Solid-state drives (SSD) have no moving parts. This reduces random access time and read latency while accelerating data throughput. Because their price per 1MB is significantly higher than the price of spinning hard disks, SSDs are only suitable for smaller storage.

OSDs may see a significant performance improvement by storing their WAL/DB partitions on an SSD and the object data on a separate hard disk.



### Tip: Sharing an SSD for Multiple WAL/DB Partitions

As WAL/DB partitions occupy relatively little space, you can share one SSD disk with multiple WAL/DB partitions. Keep in mind that with each WAL/DB partition, the performance of the SSD disk degrades. We do not recommend sharing more than six WAL/DB partitions on the same SSD disk and 12 on NVMe disks.

## 2.4.5 Maximum recommended number of disks

You can have as many disks in one server as it allows. There are a few things to consider when planning the number of disks per server:

- *Network bandwidth.* The more disks you have in a server, the more data must be transferred via the network card(s) for the disk write operations.
- *Memory.* RAM above 2 GB is used for the BlueStore cache. With the default `osd_memory_target` of 4 GB, the system has a reasonable starting cache size for spinning media. If using SSD or NVME, consider increasing the cache size and RAM allocation per OSD to maximize performance.
- *Fault tolerance.* If the complete server fails, the more disks it has, the more OSDs the cluster temporarily loses. Moreover, to keep the replication rules running, you need to copy all the data from the failed server among the other nodes in the cluster.

## 2.5 Monitor nodes

- At least three MON nodes are required. The number of monitors should always be odd ( $1 + 2n$ ).
- 4 GB of RAM.
- Processor with four logical cores.
- An SSD or other sufficiently fast storage type is highly recommended for monitors, specifically for the `/var/lib/ceph` path on each monitor node, as quorum may be unstable with high disk latencies. Two disks in RAID 1 configuration is recommended for redundancy. It is recommended that separate disks or at least separate disk partitions are used for the monitor processes to protect the monitor's available disk space from things like log file creep.
- There must only be one monitor process per node.
- Mixing OSD, MON, or Object Gateway nodes is only supported if sufficient hardware resources are available. That means that the requirements for all services need to be added up.
- Two network interfaces bonded to multiple switches.

## 2.6 Object Gateway nodes

Object Gateway nodes should have at least six CPU cores and 32 GB of RAM. When other processes are co-located on the same machine, their requirements need to be added up.

## 2.7 Metadata Server nodes

Proper sizing of the Metadata Server nodes depends on the specific use case. Generally, the more open files the Metadata Server is to handle, the more CPU and RAM it needs. The following are the minimum requirements:

- 4 GB of RAM for each Metadata Server daemon.
- Bonded network interface.
- 2.5 GHz CPU with at least 2 cores.

## 2.8 Admin Node

At least 4 GB of RAM and a quad-core CPU are required. This includes running the Salt Master on the Admin Node. For large clusters with hundreds of nodes, 6 GB of RAM is suggested.

## 2.9 iSCSI Gateway nodes

iSCSI Gateway nodes should have at least six CPU cores and 16 GB of RAM.

## 2.10 SES and other SUSE products

This section contains important information about integrating SES with other SUSE products.

### 2.10.1 SUSE Manager

SUSE Manager and SUSE Enterprise Storage are not integrated, therefore SUSE Manager cannot currently manage an SES cluster.

## 2.11 Name limitations

Ceph does not generally support non-ASCII characters in configuration files, pool names, user names and so forth. When configuring a Ceph cluster we recommend using only simple alphanumeric characters (A-Z, a-z, 0-9) and minimal punctuation (., ', \_) in all Ceph object/configuration names.

## 2.12 OSD and monitor sharing one server

Although it is technically possible to run OSDs and MONs on the same server in test environments, we strongly recommend having a separate server for each monitor node in production. The main reason is performance—the more OSDs the cluster has, the more I/O operations the MON nodes need to perform. And when one server is shared between a MON node and OSD(s), the OSD I/O operations are a limiting factor for the monitor node.

Another consideration is whether to share disks between an OSD, a MON node, and the operating system on the server. The answer is simple: if possible, dedicate a separate disk to OSD, and a separate server to a monitor node.

Although Ceph supports directory-based OSDs, an OSD should always have a dedicated disk other than the operating system one.



### Tip

If it is *really* necessary to run OSD and MON node on the same server, run MON on a separate disk by mounting the disk to the `/var/lib/ceph/mon` directory for slightly better performance.

## 3 Admin Node HA setup

The *Admin Node* is a Ceph cluster node where the Salt Master service runs. It manages the rest of the cluster nodes by querying and instructing their Salt Minion services. It usually includes other services as well, for example the *Grafana* dashboard backed by the *Prometheus* monitoring toolkit.

In case of Admin Node failure, you usually need to provide new working hardware for the node and restore the complete cluster configuration stack from a recent backup. Such a method is time consuming and causes cluster outage.

To prevent the Ceph cluster performance downtime caused by the Admin Node failure, we recommend making use of a High Availability (HA) cluster for the Ceph Admin Node.

### 3.1 Outline of the HA cluster for Admin Node

The idea of an HA cluster is that in case of one cluster node failing, the other node automatically takes over its role, including the virtualized Admin Node. This way, other Ceph cluster nodes do not notice that the Admin Node failed.

The minimal HA solution for the Admin Node requires the following hardware:

- Two bare metal servers able to run SUSE Linux Enterprise with the High Availability extension and virtualize the Admin Node.
- Two or more redundant network communication paths, for example via Network Device Bonding.
- Shared storage to host the disk image(s) of the Admin Node virtual machine. The shared storage needs to be accessible from both servers. It can be, for example, an NFS export, a Samba share, or iSCSI target.

Find more details on the cluster requirements at <https://documentation.suse.com/sle-ha/15-SP2/html/SLE-HA-all/art-sleha-install-quick.html#sec-ha-inst-quick-req>.

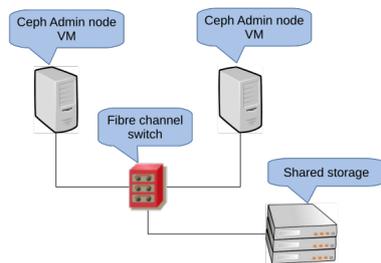


FIGURE 3.1: 2-NODE HA CLUSTER FOR ADMIN NODE

## 3.2 Building an HA cluster with the Admin Node

The following procedure summarizes the most important steps of building the HA cluster for virtualizing the Admin Node. For details, refer to the indicated links.

1. Set up a basic 2-node HA cluster with shared storage as described in <https://documentation.suse.com/sle-ha/15-SP2/html/SLE-HA-all/art-sleha-install-quick.html>.
2. On both cluster nodes, install all packages required for running the KVM hypervisor and the `libvirt` toolkit as described in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-vt-installation.html#sec-vt-installation-kvm>.
3. On the first cluster node, create a new KVM virtual machine (VM) making use of `libvirt` as described in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-kvm-inst.html#sec-libvirt-inst-virt-install>. Use the preconfigured shared storage to store the disk images of the VM.
4. After the VM setup is complete, export its configuration to an XML file on the shared storage. Use the following syntax:

```
# virsh dumpxml VM_NAME > /path/to/shared/vm_name.xml
```

5. Create a resource for the Admin Node VM. Refer to <https://documentation.suse.com/sle-ha/15-SP2/html/SLE-HA-all/cha-conf-hawk2.html> for general info on creating HA resources. Detailed info on creating resources for a KVM virtual machine is described in [http://www.linux-ha.org/wiki/VirtualDomain\\_%28resource\\_agent%29](http://www.linux-ha.org/wiki/VirtualDomain_%28resource_agent%29).
6. On the newly-created VM guest, deploy the Admin Node including the additional services you need there. Follow the relevant steps in *Chapter 6, Deploying Salt*. At the same time, deploy the remaining Ceph cluster nodes on the non-HA cluster servers.

## II Deploying Ceph Cluster

- 4 Introduction and common tasks **28**
- 5 Installing and configuring SUSE Linux Enterprise Server **30**
- 6 Deploying Salt **31**
- 7 Deploying the bootstrap cluster using `ceph-salt` **34**
- 8 Deploying the remaining core services using `cephadm` **55**
- 9 Deployment of additional services **71**

## 4 Introduction and common tasks

Since SUSE Enterprise Storage 7, Ceph services are deployed as containers instead of RPM packages. The deployment process has two basic steps:

### Deploying bootstrap cluster

This phase is called *Day 1 deployment* and consists of the following tasks: It includes installing the underlying operating system, configuring the Salt infrastructure, and deploying the minimal cluster that consist of one MON and one MGR service.

- Install and do basic configuration of the underlying operating system—SUSE Linux Enterprise Server 15 SP2—on all cluster nodes.
- Deploy the Salt infrastructure on all cluster nodes for performing the initial deployment preparations via `ceph-salt`.
- Configure the basic properties of the cluster via `ceph-salt` and deploy it.

### Deploying additional services

During *Day 2 deployment*, additional core and non-core Ceph services, for example gateways and monitoring stack, are deployed.



### Important

Note that the Ceph community documentation uses the `cephadm bootstrap` command during initial deployment. `ceph-salt` calls the `cephadm bootstrap` command automatically. The `cephadm bootstrap` command should not be run directly. Any Ceph cluster deployment manually using the `cephadm bootstrap` will be unsupported.

## 4.1 Read the release notes

In the release notes you can find additional information on changes since the previous release of SUSE Enterprise Storage. Check the release notes to see whether:

- your hardware needs special considerations.
- any used software packages have changed significantly.
- special precautions are necessary for your installation.

The release notes also provide information that could not make it into the manual on time. They also contain notes about known issues.

After having installed the package `release-notes-ses`, find the release notes locally in the directory `/usr/share/doc/release-notes` or online at <https://www.suse.com/releasenotes/>.

## 5 Installing and configuring SUSE Linux Enterprise Server

1. Install and register SUSE Linux Enterprise Server 15 SP2 on each cluster node. During installation of SUSE Enterprise Storage, access to the update repositories is required, therefore registration is mandatory. Include at least the following modules:

- Basesystem Module
- Server Applications Module

Find more details on how to install SUSE Linux Enterprise Server in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-install.html>.

2. Install the *SUSE Enterprise Storage 7* extension on each cluster node.



### Tip: Install SUSE Enterprise Storage together with SUSE Linux Enterprise Server

You can either install the SUSE Enterprise Storage 7 extension separately after you have installed SUSE Linux Enterprise Server 15 SP2, or you can add it during the SUSE Linux Enterprise Server 15 SP2 installation procedure.

Find more details on how to install extensions in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-register-sle.html>.

3. Configure network settings including proper DNS name resolution on each node. For more information on configuring a network, see <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-network.html#sec-network-yast>. For more information on configuring a DNS server, see <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-dns.html>.

## 6 Deploying Salt

SUSE Enterprise Storage uses Salt and `ceph-salt` for the initial cluster preparation. Salt helps you configure and run commands on multiple cluster nodes simultaneously from one dedicated host called the *Salt Master*. Before deploying Salt, consider the following important points:

- *Salt Minions* are the nodes controlled by a dedicated node called Salt Master.
- If the Salt Master host should be part of the Ceph cluster, it needs to run its own Salt Minion, but this is not a requirement.



### Tip: Sharing multiple roles per server

You will get the best performance from your Ceph cluster when each role is deployed on a separate node. But real deployments sometimes require sharing one node for multiple roles. To avoid trouble with performance and the upgrade procedure, do not deploy the Ceph OSD, Metadata Server, or Ceph Monitor role to the Admin Node.

- Salt Minions need to correctly resolve the Salt Master's host name over the network. By default, they look for the `salt` host name, but you can specify any other network-reachable host name in the `/etc/salt/minion` file.

1. Install the `salt-master` on the Salt Master node:

```
root@master # zypper in salt-master
```

2. Check that the `salt-master` service is enabled and started, and enable and start it if needed:

```
root@master # systemctl enable salt-master.service
root@master # systemctl start salt-master.service
```

3. If you intend to use the firewall, verify that the Salt Master node has ports 4505 and 4506 open to all Salt Minion nodes. If the ports are closed, you can open them using the `yast2 firewall` command by allowing the `salt-master` service for the appropriate zone. For example, `public`.

4. Install the package `salt-minion` on all minion nodes.

```
root@minion > zypper in salt-minion
```

5. Edit `/etc/salt/minion` and uncomment the following line:

```
#log_level_logfile: warning
```

Change the `warning` log level to `info`.



### Note: `log_level_logfile` and `log_level`

While `log_level` controls which log messages will be displayed on the screen, `log_level_logfile` controls which log messages will be written to `/var/log/salt/minion`.



### Note

Ensure you change the log level on *all* cluster (minion) nodes.

6. Make sure that the *fully qualified domain name* of each node can be resolved to an IP address on the public cluster network by all the other nodes.
7. Configure all minions to connect to the master. If your Salt Master is not reachable by the host name `salt`, edit the file `/etc/salt/minion` or create a new file `/etc/salt/minion.d/master.conf` with the following content:

```
master: host_name_of_salt_master
```

If you performed any changes to the configuration files mentioned above, restart the Salt service on all related Salt Minions:

```
root@minion > systemctl restart salt-minion.service
```

8. Check that the `salt-minion` service is enabled and started on all nodes. Enable and start it if needed:

```
# systemctl enable salt-minion.service
# systemctl start salt-minion.service
```

9. Verify each Salt Minion's fingerprint and accept all salt keys on the Salt Master if the fingerprints match.



## Note

If the Salt Minion fingerprint comes back empty, make sure the Salt Minion has a Salt Master configuration and that it can communicate with the Salt Master.

View each minion's fingerprint:

```
root@minion > salt-call --local key.finger
local:
3f:a3:2f:3f:b4:d3:d9:24:49:ca:6b:2c:e1:6c:3f:c3:83:37:f0:aa:87:42:e8:ff...
```

After gathering fingerprints of all the Salt Minions, list fingerprints of all unaccepted minion keys on the Salt Master:

```
root@master # salt-key -F
[...]
Unaccepted Keys:
minion1:
3f:a3:2f:3f:b4:d3:d9:24:49:ca:6b:2c:e1:6c:3f:c3:83:37:f0:aa:87:42:e8:ff...
```

If the minions' fingerprints match, accept them:

```
root@master # salt-key --accept-all
```

10. Verify that the keys have been accepted:

```
root@master # salt-key --list-all
```

11. Test whether all Salt Minions respond:

```
root@master # salt-run manage.status
```

## 7 Deploying the bootstrap cluster using `ceph-salt`

This section guides you through the process of deploying a basic Ceph cluster. Read the following subsections carefully and execute the included commands in the given order.

### 7.1 Installing `ceph-salt`

`ceph-salt` provides tools for deploying Ceph clusters managed by `cephadm`. `ceph-salt` uses the Salt infrastructure to perform OS management—for example, software updates or time synchronization—and defining roles for Salt Minions.

On the Salt Master, install the `ceph-salt` package:

```
root@master # zypper install ceph-salt
```

The above command installed `ceph-salt-formula` as a dependency which modified the Salt Master configuration by inserting additional files in the `/etc/salt/master.d` directory. To apply the changes, restart `salt-master.service` and synchronize Salt modules:

```
root@master # systemctl restart salt-master.service
root@master # salt \* saltutil.sync_all
```

### 7.2 Configuring cluster properties

Use the `ceph-salt config` command to configure the basic properties of the cluster.



#### Important

The `/etc/ceph/ceph.conf` file is managed by `cephadm` and users *should not* edit it. Ceph configuration parameters should be set using the new `ceph config` command. See Book “Administration and Operations Guide”, Chapter 28 “Ceph cluster configuration”, Section 28.2 “Configuration database” for more information.

## 7.2.1 Using the ceph-salt shell

If you run `ceph-salt config` without any path or subcommand, you will enter an interactive `ceph-salt` shell. The shell is convenient if you need to configure multiple properties in one batch and do not want type the full command syntax.

```
root@master # ceph-salt config
/> ls
o- / ..... [...]
  o- ceph_cluster ..... [...]
    | o- minions ..... [no minions]
    | o- roles ..... [...]
    |   o- admin ..... [no minions]
    |   o- bootstrap ..... [no minion]
    |   o- cephadm ..... [no minions]
    |   o- tuned ..... [...]
    |     o- latency ..... [no minions]
    |     o- throughput ..... [no minions]
  o- cephadm_bootstrap ..... [...]
    | o- advanced ..... [...]
    | o- ceph_conf ..... [...]
    | o- ceph_image_path ..... [ no image path]
    | o- dashboard ..... [...]
    | | o- force_password_update ..... [enabled]
    | | o- password ..... [admin]
    | | o- ssl_certificate ..... [not set]
    | | o- ssl_certificate_key ..... [not set]
    | | o- username ..... [admin]
    | o- mon_ip ..... [not set]
  o- containers ..... [...]
    | o- registries_conf ..... [enabled]
    | | o- registries ..... [empty]
    | o- registry_auth ..... [...]
    |   o- password ..... [not set]
    |   o- registry ..... [not set]
    |   o- username ..... [not set]
  o- ssh ..... [no key pair set]
    | o- private_key ..... [no private key set]
    | o- public_key ..... [no public key set]
  o- time_server ..... [enabled, no server host set]
    o- external_servers ..... [empty]
    o- servers ..... [empty]
    o- subnet ..... [not set]
```

As you can see from the output of `ceph-salt's ls` command, the cluster configuration is organized in a tree structure. To configure a specific property of the cluster in the `ceph-salt` shell, you have two options:

- Run the command from the current position and enter the absolute path to the property as the first argument:

```
/> /cephadm_bootstrap/dashboard ls
o- dashboard ..... [...]
  o- force_password_update ..... [enabled]
  o- password ..... [admin]
  o- ssl_certificate ..... [not set]
  o- ssl_certificate_key ..... [not set]
  o- username ..... [admin]
/> /cephadm_bootstrap/dashboard/username set ceph-admin
Value set.
```

- Change to the path whose property you need to configure and run the command:

```
/> cd /cephadm_bootstrap/dashboard/
/ceph_cluster/minions> ls
o- dashboard ..... [...]
  o- force_password_update ..... [enabled]
  o- password ..... [admin]
  o- ssl_certificate ..... [not set]
  o- ssl_certificate_key ..... [not set]
  o- username ..... [ceph-admin]
```



### Tip: Autocompletion of configuration snippets

While in a `ceph-salt` shell, you can use the autocompletion feature similar to a normal Linux shell (Bash) autocompletion. It completes configuration paths, subcommands, or Salt Minion names. When autocompleting a configuration path, you have two options:

- To let the shell finish a path relative to your current position, press the TAB key `-|` twice.
- To let the shell finish an absolute path, enter `/` and press the TAB key `-|` twice.



## Tip: Navigating with the cursor keys

If you enter `cd` from the `ceph-salt` shell without any path, the command will print a tree structure of the cluster configuration with the line of the current path active. You can use the up and down cursor keys to navigate through individual lines. After you confirm with `Enter`, the configuration path will change to the last active one.



## Important: Convention

To keep the documentation consistent, we will use a single command syntax without entering the `ceph-salt` shell. For example, you can list the cluster configuration tree by using the following command:

```
root@master # ceph-salt config ls
```

## 7.2.2 Adding Salt Minions

Include all or a subset of Salt Minions that we deployed and accepted in [Chapter 6, Deploying Salt](#) to the Ceph cluster configuration. You can either specify the Salt Minions by their full names, or use a glob expressions `*` and `?` to include multiple Salt Minions at once. Use the `add` subcommand under the `/ceph_cluster/minions` path. The following command includes all accepted Salt Minions:

```
root@master # ceph-salt config /ceph_cluster/minions add '*'
```

Verify that the specified Salt Minions were added:

```
root@master # ceph-salt config /ceph_cluster/minions ls
o- minions ..... [Minions: 5]
  o- ses-master.example.com ..... [no roles]
  o- ses-min1.example.com ..... [no roles]
  o- ses-min2.example.com ..... [no roles]
  o- ses-min3.example.com ..... [no roles]
  o- ses-min4.example.com ..... [no roles]
```

### 7.2.3 Specifying Salt Minions managed by cephadm

Specify which nodes will belong to the Ceph cluster and will be managed by cephadm. Include all nodes that will run Ceph services as well as the Admin Node:

```
root@master # ceph-salt config /ceph_cluster/roles/cephadm add '*'
```

### 7.2.4 Specifying Admin Node

The Admin Node is the node where the `ceph.conf` configuration file and the Ceph admin keyring is installed. You usually run Ceph related commands on the Admin Node.



#### Tip: Salt Master and Admin Node on the Same Node

In a homogeneous environment where all or most hosts belong to SUSE Enterprise Storage, we recommend having the Admin Node on the same host as the Salt Master.

In a heterogeneous environment where one Salt infrastructure hosts more than one cluster, for example, SUSE Enterprise Storage together with SUSE Manager, do *not* place the Admin Node on the same host as Salt Master.

To specify the Admin Node, run the following command:

```
root@master # ceph-salt config /ceph_cluster/roles/admin add ses-master.example.com
1 minion added.
root@master # ceph-salt config /ceph_cluster/roles/admin ls
o- admin ..... [Minions: 1]
o- ses-master.example.com ..... [Other roles: cephadm]
```



#### Tip: Install `ceph.conf` and the admin keyring on multiple nodes

You can install the Ceph configuration file and admin keyring on multiple nodes if your deployment requires it. For security reasons, avoid installing them on all the cluster's nodes.

### 7.2.5 Specifying first MON/MGR node

You need to specify which of the cluster's Salt Minions will bootstrap the cluster. This minion will become the first one running Ceph Monitor and Ceph Manager services.

```
root@master # ceph-salt config /ceph_cluster/roles/bootstrap set ses-min1.example.com
Value set.
root@master # ceph-salt config /ceph_cluster/roles/bootstrap ls
o- bootstrap ..... [ses-min1.example.com]
```

Additionally, you need to specify the bootstrap MON's IP address on the public network to ensure that the `public_network` parameter is set correctly, for example:

```
root@master # ceph-salt config /cephadm_bootstrap/mon_ip set 192.168.10.20
```

## 7.2.6 Specifying tuned profiles

You need to specify which of the cluster's minions have actively tuned profiles. To do so, add these roles explicitly with the following commands:



### Note

One minion cannot have both the `latency` and `throughput` roles.

```
root@master # ceph-salt config /ceph_cluster/roles/tuned/latency add ses-min1.example.com
Adding ses-min1.example.com...
1 minion added.
root@master # ceph-salt config /ceph_cluster/roles/tuned/throughput add ses-
min2.example.com
Adding ses-min2.example.com...
1 minion added.
```

## 7.2.7 Generating an SSH key pair

`cephadm` uses the SSH protocol to communicate with cluster nodes. A user account named `cephadm` is automatically created and used for SSH communication.

You need to generate the private and public part of the SSH key pair:

```
root@master # ceph-salt config /ssh generate
Key pair generated.
root@master # ceph-salt config /ssh ls
o- ssh ..... [Key Pair set]
o- private_key ..... [53:b1:eb:65:d2:3a:ff:51:6c:e2:1b:ca:84:8e:0e:83]
o- public_key ..... [53:b1:eb:65:d2:3a:ff:51:6c:e2:1b:ca:84:8e:0e:83]
```

## 7.2.8 Configuring the time server

All cluster nodes need to have their time synchronized with a reliable time source. There are several scenarios to approach time synchronization:

- If all cluster nodes are already configured to synchronize their time using an NTP service of choice, disable time server handling completely:

```
root@master # ceph-salt config /time_server disable
```

- If your site already has a single source of time, specify the host name of the time source:

```
root@master # ceph-salt config /time_server/servers add time-server.example.com
```

- Alternatively, `ceph-salt` has the ability to configure one of the Salt Minion to serve as the time server for the rest of the cluster. This is sometimes referred to as an "internal time server". In this scenario, `ceph-salt` will configure the internal time server (which should be one of the Salt Minion) to synchronize its time with an external time server, such as `pool.ntp.org`, and configure all the other minions to get their time from the internal time server. This can be achieved as follows:

```
root@master # ceph-salt config /time_server/servers add ses-master.example.com
root@master # ceph-salt config /time_server/external_servers add pool.ntp.org
```

The `/time_server/subnet` option specifies the subnet from which NTP clients are allowed to access the NTP server. It is automatically set when you specify `/time_server/servers`. If you need to change it or specify it manually, run:

```
root@master # ceph-salt config /time_server/subnet set 10.20.6.0/24
```

Check the time server settings:

```
root@master # ceph-salt config /time_server ls
o- time_server ..... [enabled]
  o- external_servers ..... [1]
    | o- pool.ntp.org ..... [...]
  o- servers ..... [1]
    | o- ses-master.example.com ..... [...]
  o- subnet ..... [10.20.6.0/24]
```

Find more information on setting up time synchronization in <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-ntp.html#sec-ntp-yast>.

## 7.2.9 Configuring the Ceph Dashboard login credentials

Ceph Dashboard will be available after the basic cluster is deployed. To access it, you need to set a valid user name and password, for example:

```
root@master # ceph-salt config /cephadm_bootstrap/dashboard/username set admin
root@master # ceph-salt config /cephadm_bootstrap/dashboard/password set PwD
```



### Tip: Forcing password update

By default, the first dashboard user will be forced to change their password on first login to the dashboard. To disable this feature, run the following command:

```
root@master # ceph-salt config /cephadm_bootstrap/dashboard/force_password_update
disable
```

## 7.2.10 Using the container registry

The Ceph cluster needs to have access to a container registry so that it can download and deploy containerized Ceph services. There are two ways to access the registry:

- If your cluster can access the default registry at [registry.suse.com](https://registry.suse.com) (directly or via proxy), you can point `ceph-salt` directly to this URL without creating a local registry. Continue by following the steps in [Section 7.2.10.2, “Configuring the path to container images”](#).
- If your cluster cannot access the default registry—for example, for an air-gapped deployment—you need to configure a local container registry. After the local registry is created and configured, you need to point `ceph-salt` to it.

### 7.2.10.1 Creating and configuring the local registry (optional)



#### Important

There are numerous methods of creating a local registry. The instructions in this section are examples of creating secure and insecure registries. For general information on running a container image registry, refer to <https://documentation.suse.com/sles/15-SP2/html/SLES-all/cha-registry-installation.html#sec-docker-registry-installation>.



## Tip: Placement and port usage

Deploy the registry on a machine accessible by all nodes in the cluster. We recommend the Admin Node. By default, the registry listens on port 5000.

On the registry node, use the following command to ensure that the port is free:

```
ss -tulpn | grep :5000
```

If other processes (such as `iscsi-tcmu`) are already listening on port 5000, determine another free port which can be used to map to port 5000 in the registry container.

### PROCEDURE 7.1: CREATING THE LOCAL REGISTRY

1. Verify that the `Containers Module` extension is enabled:

```
> SUSEConnect --list-extensions | grep -A2 "Containers Module"
Containers Module 15 SP2 x86_64 (Activated)
```

2. Verify that the following packages are installed: `apache2-utils` (if enabling a secure registry), `cni`, `cni-plugins`, `podman`, `podman-cni-config`, and `skopeo`.

3. Gather the following information:

- Fully qualified domain name of the registry host (`REG_HOST_FQDN`).
- An available port number used to map to the registry container port of 5000 (`REG_HOST_PORT`).
- Whether the registry will be secure or insecure (`insecure=[true|false]`).

4. To start an insecure registry (without SSL encryption), follow these steps:

- a. Configure `ceph-salt` for the insecure registry:

```
cephuser@adm > ceph-salt config containers/registries_conf enable
cephuser@adm > ceph-salt config containers/registries_conf/registries \
  add prefix=REG_HOST_FQDN insecure=true \
  location=REG_HOST_PORT:5000
```

- b. Start the insecure registry by creating the necessary directory (for example, `/var/lib/registry`) and starting the registry with the `podman` command:

```
# mkdir -p /var/lib/registry
# podman run --privileged -d --name registry \
```

```
-p REG_HOST_PORT:5000 -v /var/lib/registry:/var/lib/registry \  
--restart=always registry:2
```

- c. To have the registry start after a reboot, create a `systemd` unit file for it and enable it:

```
> sudo podman generate systemd --files --name registry  
> sudo mv container-registry.service /etc/systemd/system/  
> sudo systemctl enable container-registry.service
```

5. To start a secure registry, follow these steps:

- a. Create the necessary directories:

```
# mkdir -p /var/lib/registry/{auth,certs}
```

- b. Generate an SSL certificate:

```
# openssl req -newkey rsa:4096 -nodes -sha256 \  
-keyout /var/lib/registry/certs/domain.key -x509 -days 365 \  
-out /var/lib/registry/certs/domain.crt
```



## Note

Set the `CN=[value]` value to the fully qualified domain name of the host (`[REG_HOST_FQDN]`).

- c. Copy the certificate to all cluster nodes and refresh the certificate cache:

```
# salt-cp '*' /var/lib/registry/certs/domain.crt \  
/etc/pki/trust/anchors/  
# salt '*' cmd.shell "update-ca-certificates"
```

- d. Generate a username and password combination for authentication to the registry:

```
# htpasswd2 -bBc /var/lib/registry/auth/htpasswd \  
REG_USERNAME REG_PASSWORD
```

- e. Start the secure registry. Use the `REGISTRY_STORAGE_DELETE_ENABLED=true` flag so that you can delete images afterwards with the `skopeo delete` command.

```
podman run --name myregistry -p REG_HOST_PORT:5000 \  
-v /var/lib/registry:/var/lib/registry \  
-v /var/lib/registry/auth:/auth:z \  
-e "REGISTRY_AUTH=htpasswd" \  
-e "REGISTRY_STORAGE_DELETE_ENABLED=true"
```

```
-e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \  
-e REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd \  
-v /var/lib/registry/certs:/certs:z \  
-e "REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt" \  
-e "REGISTRY_HTTP_TLS_KEY=/certs/domain.key" \  
-e REGISTRY_STORAGE_DELETE_ENABLED=true \  
-e REGISTRY_COMPATIBILITY_SCHEMA1_ENABLED=true -d registry:2
```

f. Test secure access to the registry:

```
> curl https://REG_HOST_FQDN:REG_HOST_PORT/v2/_catalog \  
-u REG_USERNAME:REG_PASSWORD
```

6. When the local registry is created, you need to synchronize container images from the official SUSE registry at [registry.suse.com](https://registry.suse.com) to the local one. You can use the **skopeo sync** command found in the **skopeo** package for that purpose. For more details, refer to the manual page (**man 1 skopeo-sync**). Consider the following examples:

#### EXAMPLE 7.1: VIEWING MANIFEST FILES

```
skopeo inspect docker://registry.suse.com/ses/7/ceph/ceph | jq .RepoTags  
skopeo inspect docker://registry.suse.com/ses/7/ceph/grafana | jq .RepoTags  
skopeo inspect docker://registry.suse.com/ses/7/ceph/prometheus-server:2.27.1 |  
jq .RepoTags  
skopeo inspect docker://registry.suse.com/ses/7/ceph/prometheus-node-exporter:1.1.2  
| jq .RepoTags  
skopeo inspect docker://registry.suse.com/ses/7/ceph/prometheus-alertmanager:0.21.0  
| jq .RepoTags
```

#### EXAMPLE 7.2: SYNCHRONIZE TO A DIRECTORY

Synchronize all Ceph images:

```
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/ceph /root/  
images/
```

Synchronize just the latest images:

```
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/ceph:latest /  
root/images/
```

#### EXAMPLE 7.3: SYNCHRONIZE GRAFANA IMAGES:

```
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/grafana /  
root/images/
```

Synchronize the latest Grafana images only:

```
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/
grafana:latest /root/images/
```

#### EXAMPLE 7.4: SYNCHRONIZE LATEST PROMETHEUS IMAGES

```
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/prometheus-
server:2.27.1 /root/images/
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/prometheus-node-
exporter:1.1.2 /root/images/
skopeo sync --src docker --dest dir registry.suse.com/ses/7/ceph/prometheus-
alertmanager:0.21.0 /root/images/
```

#### PROCEDURE 7.2: CONFIGURE THE LOCAL REGISTRY AND ACCESS CREDENTIALS

1. Configure the URL of the local registry:

```
cephuser@adm > ceph-salt config /containers/registry_auth/registry set REG_HOST_URL
```

2. Configure the user name and password to access the local registry:

```
cephuser@adm > ceph-salt config /containers/registry_auth/username set REG_USERNAME
```

```
cephuser@adm > ceph-salt config /containers/registry_auth/password set REG_PASSWORD
```



#### Tip: Registry cache

To avoid re-syncing the local registry when new updated containers appear, you can configure a *registry cache*.

#### 7.2.10.2 Configuring the path to container images



#### Important

This section helps you configure the path to container images of the bootstrap cluster (deployment of the first Ceph Monitor and Ceph Manager pair). The path does not apply to container images of additional services, for example the monitoring stack.



## Tip: Configuring HTTPS proxy

If you need to use a proxy to communicate with the container registry server, perform the following configuration steps on all cluster nodes:

1. Copy the configuration file for containers:

```
> sudo cp /usr/share/containers/containers.conf /etc/containers/containers.conf
```

2. Edit the newly-copied file and add the `http_proxy` setting to its `[engine]` section, for example:

```
> cat /etc/containers/containers.conf
[engine]
http_proxy=proxy.example.com
[...]
```

`cephadm` needs to know a valid URI path to container images. Verify the default setting by executing

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_image_path ls
```

If you do not need an alternative or local registry, specify the default SUSE container registry:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_image_path set registry.suse.com/ses/7/ceph/ceph
```

If your deployment requires a specific path, for example, a path to a local registry, configure it as follows:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_image_path set LOCAL_REGISTRY_PATH
```

### 7.2.11 Enabling data in-flight encryption (msgr2)

The Messenger v2 protocol (MSGR2) is Ceph's on-wire protocol. It provides a security mode that encrypts all data passing over the network, encapsulation of authentication payloads, and the enabling of future integration of new authentication modes (such as Kerberos).



## Important

`msgsr2` is not currently supported by Linux kernel Ceph clients, such as CephFS and RADOS Block Device.

Ceph daemons can bind to multiple ports, allowing both legacy Ceph clients and new v2-capable clients to connect to the same cluster. By default, MONs now bind to the new IANA-assigned port 3300 (CE4h or 0xCE4) for the new v2 protocol, while also binding to the old default port 6789 for the legacy v1 protocol.

The v2 protocol (MSGSR2) supports two connection modes:

### `crc mode`

A strong initial authentication when the connection is established and a CRC32C integrity check.

### `secure mode`

A strong initial authentication when the connection is established and full encryption of all post-authentication traffic, including a cryptographic integrity check.

For most connections, there are options that control which modes are used:

### `ms_cluster_mode`

The connection mode (or permitted modes) used for intra-cluster communication between Ceph daemons. If multiple modes are listed, the modes listed first are preferred.

### `ms_service_mode`

A list of permitted modes for clients to use when connecting to the cluster.

### `ms_client_mode`

A list of connection modes, in order of preference, for clients to use (or allow) when talking to a Ceph cluster.

There are a parallel set of options that apply specifically to monitors, allowing administrators to set different (usually more secure) requirements on communication with the monitors.

### `ms_mon_cluster_mode`

The connection mode (or permitted modes) to use between monitors.

### `ms_mon_service_mode`

A list of permitted modes for clients or other Ceph daemons to use when connecting to monitors.

## ms\_mon\_client\_mode

A list of connection modes, in order of preference, for clients or non-monitor daemons to use when connecting to monitors.

In order to enable MSGR2 encryption mode during the deployment, you need to add some configuration options to the `ceph-salt` configuration before running `ceph-salt apply`.

To use `secure` mode, run the following commands.

Add the global section to `ceph_conf` in the `ceph-salt` configuration tool:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf add global
```

Set the following options:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_cluster_mode "secure crc"
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_service_mode "secure crc"
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_client_mode "secure crc"
```



## Note

Ensure `secure` precedes `crc`.

To *force* `secure` mode, run the following commands:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_cluster_mode secure
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_service_mode secure
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set ms_client_mode secure
```



## Tip: Updating settings

If you want to change any of the above settings, set the configuration changes in the monitor configuration store. This is achieved using the `ceph config set` command.

```
root@master # ceph config set global CONNECTION_OPTION CONNECTION_MODE [--force]
```

For example:

```
root@master # ceph config set global ms_cluster_mode "secure crc"
```

If you want to check the current value, including default value, run the following command:

```
root@master # ceph config get CEPH_COMPONENT CONNECTION_OPTION
```

For example, to get the `ms_cluster_mode` for OSD's, run:

```
root@master # ceph config get osd ms_cluster_mode
```

## 7.2.12 Configuring the cluster network

Optionally, if you are running a separate cluster network, you may need to set the cluster network IP address followed by the subnet mask part after the slash sign, for example 192.168.10.22/24.

Run the following commands to enable `cluster_network`:

```
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf add global
root@master # ceph-salt config /cephadm_bootstrap/ceph_conf/global set
cluster_network NETWORK_ADDR
```

## 7.2.13 Verifying the cluster configuration

The minimal cluster configuration is finished. Inspect it for obvious errors:

```
root@master # ceph-salt config ls
o- / ..... [...]
  o- ceph_cluster ..... [...]
    | o- minions ..... [Minions: 5]
    | | o- ses-master.example.com ..... [admin]
    | | o- ses-min1.example.com ..... [bootstrap, admin]
    | | o- ses-min2.example.com ..... [no roles]
    | | o- ses-min3.example.com ..... [no roles]
    | | o- ses-min4.example.com ..... [no roles]
    | o- roles ..... [...]
    | o- admin ..... [Minions: 2]
```

```

| | o- ses-master.example.com ..... [no other roles]
| | o- ses-min1.example.com ..... [other roles: bootstrap]
| o- bootstrap ..... [ses-min1.example.com]
| o- cephadm ..... [Minions: 5]
| o- tuned ..... [...]
|   o- latency ..... [no minions]
|   o- throughput ..... [no minions]
o- cephadm_bootstrap ..... [...]
| o- advanced ..... [...]
| o- ceph_conf ..... [...]
| o- ceph_image_path ..... [registry.suse.com/ses/7/ceph/ceph]
| o- dashboard ..... [...]
|   o- force_password_update ..... [enabled]
|   o- password ..... [randomly generated]
|   o- username ..... [admin]
| o- mon_ip ..... [192.168.10.20]
o- containers ..... [...]
| o- registries_conf ..... [enabled]
| | o- registries ..... [empty]
| o- registry_auth ..... [...]
|   o- password ..... [not set]
|   o- registry ..... [not set]
|   o- username ..... [not set]
o- ssh ..... [Key Pair set]
| o- private_key ..... [53:b1:eb:65:d2:3a:ff:51:6c:e2:1b:ca:84:8e:0e:83]
| o- public_key ..... [53:b1:eb:65:d2:3a:ff:51:6c:e2:1b:ca:84:8e:0e:83]
o- time_server ..... [enabled]
  o- external_servers ..... [1]
    | o- 0.pt.pool.ntp.org ..... [...]
  o- servers ..... [1]
    | o- ses-master.example.com ..... [...]
  o- subnet ..... [10.20.6.0/24]

```



## Tip: Status of cluster configuration

You can check if the configuration of the cluster is valid by running the following command:

```

root@master # ceph-salt status
cluster: 5 minions, 0 hosts managed by cephadm
config: OK

```

## 7.2.14 Exporting cluster configurations

After you have configured the basic cluster and its configuration is valid, it is a good idea to export its configuration to a file:

```
root@master # ceph-salt export > cluster.json
```



### Warning

The output of the **ceph-salt export** includes the SSH private key. If you are concerned about the security implications, do not execute this command without taking appropriate precautions.

In case you break the cluster configuration and need to revert to a backup state, run:

```
root@master # ceph-salt import cluster.json
```

## 7.3 Updating nodes and bootstrap minimal cluster

Before you deploy the cluster, update all software packages on all nodes:

```
root@master # ceph-salt update
```

If a node reports Reboot is needed during the update, important OS packages—such as the kernel—were updated to a newer version and you need to reboot the node to apply the changes.

To reboot all nodes that require rebooting, either append the --reboot option

```
root@master # ceph-salt update --reboot
```

Or, reboot them in a separate step:

```
root@master # ceph-salt reboot
```



### Important

The Salt Master is never rebooted by **ceph-salt update --reboot** or **ceph-salt reboot** commands. If the Salt Master needs rebooting, you need to reboot it manually.

After the nodes are updated, bootstrap the minimal cluster:

```
root@master # ceph-salt apply
```



## Note

When bootstrapping is complete, the cluster will have one Ceph Monitor and one Ceph Manager.

The above command will open an interactive user interface that shows the current progress of each minion.

```
Running: salt -G 'ceph-saltmember' state.apply ceph-salt Finished: 0/5 Duration: 191s
+ doc-ses-master.suse.cz ..... (191s)
  | _ Install and update required packages ..... (185s)
  | _ Update all packages ..... (124s)
+ doc-ses-min1.suse.cz ..... (191s)
  | _ Install and update required packages ..... (187s)
  | _ Update all packages ..... (95s)
+ doc-ses-min2.suse.cz ..... (191s)
  | _ Install and update required packages ..... (187s)
  | _ Update all packages ..... (97s)
+ doc-ses-min3.suse.cz ..... (191s)
  | _ Install and update required packages ..... (187s)
  | _ Update all packages ..... (95s)
+ doc-ses-min4.suse.cz ..... (191s)
  | _ Install and update required packages ..... (187s)
  | _ Update all packages ..... (98s)
```

FIGURE 7.1: DEPLOYMENT OF A MINIMAL CLUSTER



## Tip: Non-interactive mode

If you need to apply the configuration from a script, there is also a non-interactive mode of deployment. This is also useful when deploying the cluster from a remote machine because constant updating of the progress information on the screen over the network may become distracting:

```
root@master # ceph-salt apply --non-interactive
```

## 7.4 Reviewing final steps

After the `ceph-salt apply` command has completed, you should have one Ceph Monitor and one Ceph Manager. You should be able to run the `ceph status` command successfully on any of the minions that were given the `admin` role as `root` or the `cephadm` user using `sudo`.

The next steps involve using the `cephadm` to deploy additional Ceph Monitor, Ceph Manager, OSDs, the Monitoring Stack, and Gateways.

Before you continue, review your new cluster's network settings. At this point, the `public_network` setting has been populated based on what was entered for `/cephadm_bootstrap/mon_ip` in the `ceph-salt` configuration. However, this setting was only applied to Ceph Monitor. You can review this setting with the following command:

```
root@master # ceph config get mon public_network
```

This is the minimum that Ceph requires to work, but we recommend making this `public_network` setting `global`, which means it will apply to all types of Ceph daemons, and not only to MONs:

```
root@master # ceph config set global public_network "$(ceph config get mon public_network)"
```



## Note

This step is not required. However, if you do not use this setting, the Ceph OSDs and other daemons (except Ceph Monitor) will listen on *all addresses*.

If you want your OSDs to communicate amongst themselves using a completely separate network, run the following command:

```
root@master # ceph config set global cluster_network "cluster_network_in_cidr_notation"
```

Executing this command will ensure that the OSDs created in your deployment will use the intended cluster network from the start.

If your cluster is set to have dense nodes (greater than 62 OSDs per host), make sure to assign sufficient ports for Ceph OSDs. The default range (6800-7300) currently allows for no more than 62 OSDs per host. For a cluster with dense nodes, adjust the setting `ms_bind_port_max` to a suitable value. Each OSD will consume eight additional ports. For example, given a host that is set to run 96 OSDs, 768 ports will be needed. `ms_bind_port_max` should be set at least to 7568 by running the following command:

```
root@master # ceph config set osd.* ms_bind_port_max 7568
```

You will need to adjust your firewall settings accordingly for this to work. See *Book "Troubleshooting Guide", Chapter 13 "Hints and tips", Section 13.7 "Firewall settings for Ceph"* for more information.

## 7.5 Disable insecure clients

Since Octopus v15.2.11, a new health warning was introduced that informs you that insecure clients are allowed to join the cluster. This warning is *on* by default. The Ceph Dashboard will show the cluster in the `HEALTH_WARN` status and verifying the cluster status on the command line informs you as follows:

```
cephuser@adm > ceph status
cluster:
  id:      3fe8b35a-689f-4970-819d-0e6b11f6707c
  health: HEALTH_WARN
  mons are allowing insecure global_id reclaim
[...]
```

This warning means that the Ceph Monitors are still allowing old, unpatched clients to connect to the cluster. This ensures existing clients can still connect while the cluster is being upgraded, but warns you that there is a problem that needs to be addressed. When the cluster and all clients are upgraded to the latest version of Ceph, disallow unpatched clients by running the following command:

```
cephuser@adm > ceph config set mon auth_allow_insecure_global_id_reclaim false
```

## 8 Deploying the remaining core services using cephadm

After deploying the basic Ceph cluster, deploy core services to more cluster nodes. To make the cluster data accessible to clients, deploy additional services as well.

Currently, we support deployment of Ceph services on the command line by using the Ceph orchestrator (`ceph orch` subcommands).

### 8.1 The `ceph orch` command

The Ceph orchestrator command `ceph orch`—which is an interface to the `cephadm` module—will take care of listing cluster components and deploying Ceph services on new cluster nodes.

#### 8.1.1 Displaying the orchestrator status

The following command shows the current mode and status of the Ceph orchestrator.

```
cephuser@adm > ceph orch status
```

#### 8.1.2 Listing devices, services, and daemons

To list all disk devices, run the following:

```
cephuser@adm > ceph orch device ls
Hostname Path      Type  Serial  Size  Health  Ident  Fault  Available
ses-master /dev/vdb hdd    0d8a... 10.7G Unknown N/A    N/A    No
ses-min1   /dev/vdc hdd    8304... 10.7G Unknown N/A    N/A    No
ses-min1   /dev/vdd hdd    7b81... 10.7G Unknown N/A    N/A    No
[...]
```



#### Tip: Services and daemons

*Service* is a general term for a Ceph service of a specific type, for example Ceph Manager.

*Daemon* is a specific instance of a service, for example a process `mgr.ses-min1.gd1cik` running on a node called `ses-min1`.

To list all services known to cephadm, run:

```
cephuser@adm > ceph orch ls
NAME    RUNNING  REFRESHED  AGE  PLACEMENT  IMAGE NAME                               IMAGE ID
mgr     1/0      5m ago     -    <no spec>  registry.example.com/[...]             5bf12403d0bd
mon     1/0      5m ago     -    <no spec>  registry.example.com/[...]             5bf12403d0bd
```



## Tip

You can limit the list to services on a particular node with the optional `--host` parameter, and services of a particular type with the optional `--service-type` parameter. Acceptable types are `mon`, `osd`, `mgr`, `mds`, and `rgw`.

To list all running daemons deployed by cephadm, run:

```
cephuser@adm > ceph orch ps
NAME           HOST      STATUS  REFRESHED  AGE  VERSION  IMAGE ID      CONTAINER ID
mgr.ses-min1.gd ses-min1  running) 8m ago     12d 15.2.0.108 5bf12403d0bd b8104e09814c
mon.ses-min1    ses-min1  running) 8m ago     12d 15.2.0.108 5bf12403d0bd a719e0087369
```



## Tip

To query the status of a particular daemon, use `--daemon_type` and `--daemon_id`. For OSDs, the ID is the numeric OSD ID. For MDS, the ID is the file system name:

```
cephuser@adm > ceph orch ps --daemon_type osd --daemon_id 0
cephuser@adm > ceph orch ps --daemon_type mds --daemon_id my_cephfs
```

## 8.2 Service and placement specification

The recommended way to specify the deployment of Ceph services is to create a YAML-formatted file with the specification of the services that you intend to deploy.

### 8.2.1 Creating service specifications

You can create a separate specification file for each type of service, for example:

```
root@master # cat nfs.yml
```

```

service_type: nfs
service_id: EXAMPLE_NFS
placement:
  hosts:
    - ses-min1
    - ses-min2
spec:
  pool: EXAMPLE_POOL
  namespace: EXAMPLE_NAMESPACE

```

Alternatively, you can specify multiple (or all) service types in one file—for example, `cluster.yml`—that describes which nodes will run specific services. Remember to separate individual service types with three dashes (`---`):

```

cephuser@adm > cat cluster.yml
service_type: nfs
service_id: EXAMPLE_NFS
placement:
  hosts:
    - ses-min1
    - ses-min2
spec:
  pool: EXAMPLE_POOL
  namespace: EXAMPLE_NAMESPACE
---
service_type: rgw
service_id: REALM_NAME.ZONE_NAME
placement:
  hosts:
    - ses-min1
    - ses-min2
    - ses-min3
---
[...]
```

The aforementioned properties have the following meaning:

#### service\_type

The type of the service. It can be either a Ceph service (mon, mgr, mds, crash, osd, or rbd-mirror), a gateway (nfs or rgw), or part of the monitoring stack (alertmanager, grafana, node-exporter, or prometheus).

#### service\_id

The name of the service. Specifications of type mon, mgr, alertmanager, grafana, node-exporter, and prometheus do not require the service\_id property.

## placement

Specifies which nodes will be running the service. Refer to [Section 8.2.2, “Creating placement specification”](#) for more details.

## spec

Additional specification relevant for the service type.



### Tip: Applying specific services

Ceph cluster services have usually a number of properties specific to them. For examples and details of individual services' specification, refer to [Section 8.3, “Deploy Ceph services”](#).

## 8.2.2 Creating placement specification

To deploy Ceph services, `cephadm` needs to know on which nodes to deploy them. Use the `placement` property and list the short host names of the nodes that the service applies to:

```
cephuser@adm > cat cluster.yml
[...]
```

```
placement:
  hosts:
  - host1
  - host2
  - host3
[...]
```

## 8.2.3 Applying cluster specification

After you have created a full `cluster.yml` file with specifications of all services and their placement, you can apply the cluster by running the following command:

```
cephuser@adm > ceph orch apply -i cluster.yml
```

To view the status of the cluster, run the `ceph orch status` command. For more details, see [Section 8.1.1, “Displaying the orchestrator status”](#).

## 8.2.4 Exporting the specification of a running cluster

Although you deployed services to the Ceph cluster by using the specification files as described in [Section 8.2, “Service and placement specification”](#), the configuration of the cluster may diverge from the original specification during its operation. Also, you may have removed the specification files accidentally.

To retrieve a complete specification of a running cluster, run:

```
cephuser@adm > ceph orch ls --export
placement:
  hosts:
    - hostname: ses-min1
      name: ''
      network: ''
  service_id: my_cephfs
  service_name: mds.my_cephfs
  service_type: mds
  ---
placement:
  count: 2
  service_name: mgr
  service_type: mgr
  ---
[...]
```



### Tip

You can append the `--format` option to change the default `yaml` output format. You can select from `json`, `json-pretty`, or `yaml`. For example:

```
ceph orch ls --export --format json
```

## 8.3 Deploy Ceph services

After the basic cluster is running, you can deploy Ceph services to additional nodes.

## 8.3.1 Deploying Ceph Monitors and Ceph Managers

Ceph cluster has three or five MONs deployed across different nodes. If there are five or more nodes in the cluster, we recommend deploying five MONs. A good practice is to have MGRs deployed on the same nodes as MONs.

### Important: Include Bootstrap MON

When deploying MONs and MGRs, remember to include the first MON that you added when configuring the basic cluster in [Section 7.2.5, “Specifying first MON/MGR node”](#).

To deploy MONs, apply the following specification:

```
service_type: mon
placement:
  hosts:
  - ses-min1
  - ses-min2
  - ses-min3
```

### Note

If you need to add another node, append the host name to the same YAML list. For example:

```
service_type: mon
placement:
  hosts:
  - ses-min1
  - ses-min2
  - ses-min3
  - ses-min4
```

Similarly, to deploy MGRs, apply the following specification:

### Important

Ensure your deployment has at least three Ceph Managers in each deployment.

```
service_type: mgr
```

```
placement:
  hosts:
  - ses-min1
  - ses-min2
  - ses-min3
```



## Tip

If MONs or MGRs are *not* on the same subnet, you need to append the subnet addresses. For example:

```
service_type: mon
placement:
  hosts:
  - ses-min1:10.1.2.0/24
  - ses-min2:10.1.5.0/24
  - ses-min3:10.1.10.0/24
```

## 8.3.2 Deploying Ceph OSDs



### Important: When Storage Device is Available

A storage device is considered *available* if all of the following conditions are met:

- The device has no partitions.
- The device does not have any LVM state.
- The device is not be mounted.
- The device does not contain a file system.
- The device does not contain a BlueStore OSD.
- The device is larger than 5 GB.

If the above conditions are not met, Ceph refuses to provision such OSDs.

There are two ways you can deploy OSDs:

- Tell Ceph to consume all available and unused storage devices:

```
cephuser@adm > ceph orch apply osd --all-available-devices
```

- Use DriveGroups (see Book “Administration and Operations Guide”, Chapter 13 “Operational tasks”, Section 13.4.3 “Adding OSDs using DriveGroups specification”) to create OSD specification describing devices that will be deployed based on their properties, such as device type (SSD or HDD), device model names, size, or the nodes on which the devices exist. Then apply the specification by running the following command:

```
cephuser@adm > ceph orch apply osd -i drive_groups.yml
```

### 8.3.3 Deploying Metadata Servers

CephFS requires one or more Metadata Server (MDS) services. To create a CephFS, first create MDS servers by applying the following specification:



#### Note

Ensure you have at least two pools, one for CephFS data and one for CephFS metadata, created before applying the following specification.

```
service_type: mds
service_id: CEPHFS_NAME
placement:
  hosts:
    - ses-min1
    - ses-min2
    - ses-min3
```

After MDSs are functional, create the CephFS:

```
ceph fs new CEPHFS_NAME metadata_pool data_pool
```

### 8.3.4 Deploying Object Gateways

cephadm deploys an Object Gateway as a collection of daemons that manage a particular *realm* and *zone*.

You can either relate an Object Gateway service to already existing realm and zone, (refer to *Book "Administration and Operations Guide", Chapter 21 "Ceph Object Gateway", Section 21.13 "Multisite Object Gateways"* for more details), or you can specify a non-existing `REALM_NAME` and `ZONE_NAME` and they will be created automatically after you apply the following configuration:

```
service_type: rgw
service_id: REALM_NAME.ZONE_NAME
placement:
  hosts:
    - ses-min1
    - ses-min2
    - ses-min3
spec:
  rgw_realm: RGW_REALM
  rgw_zone: RGW_ZONE
```

#### 8.3.4.1 Using secure SSL access

To use a secure SSL connection to the Object Gateway, you need a pair of valid SSL certificate and key files (see *Book "Administration and Operations Guide", Chapter 21 "Ceph Object Gateway", Section 21.7 "Enable HTTPS/SSL for Object Gateways"* for more details). You need to enable SSL, specify a port number for SSL connections, and the SSL certificate and key files.

To enable SSL and specify the port number, include the following in your specification:

```
spec:
  ssl: true
  rgw_frontend_port: 443
```

To specify the SSL certificate and key, you can paste their contents directly into the YAML specification file. The pipe sign (`|`) at the end of line tells the parser to expect a multi-line string as a value. For example:

```
spec:
  ssl: true
  rgw_frontend_port: 443
  rgw_frontend_ssl_certificate: |
    -----BEGIN CERTIFICATE-----
    MIIIFmjCCA4KgAwIBAgIJAIZ2n35bmwXTMA0GCSqGSIb3DQEBCwUAMGIxCzAJBgNV
    BAYTAKFVMQwwCgYDVQQIDANOU1cxHTAbBgNVBAoMFEV4YW1wbGUgUkdXIFNTTCBp
    [...]
    -----END CERTIFICATE-----
  rgw_frontend_ssl_key: |
    -----BEGIN PRIVATE KEY-----
```

```
MIIJRAIBADANBgkqhkiG9w0BAQEFAASCCS4wggkqAgEAAoICAQDLtFwg6LLl2j4Z
BDV+iL4A07VZ9KbmWI+t37Ml2W6y2YeKX3Qwf+3eBz7TVHR1dm6iPpCpqpQjXUsT9
[...]
-----END PRIVATE KEY-----
```



## Tip

Instead of pasting the content of SSL certificate and key files, you can omit the `rgw_frontend_ssl_certificate:` and `rgw_frontend_ssl_key:` keywords and upload them to the configuration database:

```
cephuser@adm > ceph config-key set rgw/cert/REALM_NAME/ZONE_NAME.crt \
-i SSL_CERT_FILE
cephuser@adm > ceph config-key set rgw/cert/REALM_NAME/ZONE_NAME.key \
-i SSL_KEY_FILE
```

### 8.3.4.1.1 Configure the Object Gateway to listen on both ports 443 and 80

To configure the Object Gateway to listen on both ports 443 (HTTPS) and 80 (HTTP), follow these steps:



## Note

The commands in the procedure use `realm` and `zone` `default`.

1. Deploy the Object Gateway by supplying a specification file. Refer to [Section 8.3.4, "Deploying Object Gateways"](#) for more details on the Object Gateway specification. Use the following command:

```
cephuser@adm > ceph orch apply -i SPEC_FILE
```

2. If SSL certificates are not supplied in the specification file, add them by using the following command:

```
cephuser@adm > ceph config-key set rgw/cert/default/default.crt -i certificate.pem
cephuser@adm > ceph config-key set rgw/cert/default/default.key -i key.pem
```

3. Change the default value of the `rgw_frontends` option:

```
cephuser@adm > ceph config set client.rgw.default.default rgw_frontends \
```

```
"beast port=80 ssl_port=443"
```

#### 4. Restart Object Gateways:

```
cephuser@adm > ceph orch restart rgw.default.default
```

### 8.3.4.2 Deploying with a subcluster

*Subclusters* help you organize the nodes in your clusters to isolate workloads and make elastic scaling easier. If you are deploying with a subcluster, apply the following configuration:

```
service_type: rgw
service_id: REALM_NAME.ZONE_NAME.SUBCLUSTER
placement:
  hosts:
    - ses-min1
    - ses-min2
    - ses-min3
spec:
  rgw_realm: RGW_REALM
  rgw_zone: RGW_ZONE
  subcluster: SUBCLUSTER
```

### 8.3.5 Deploying iSCSI Gateways

cephadm deploys an iSCSI Gateway which is a storage area network (SAN) protocol that allows clients (called initiators) to send SCSI commands to SCSI storage devices (targets) on remote servers.

Apply the following configuration to deploy. Ensure `trusted_ip_list` contains the IP addresses of all iSCSI Gateway and Ceph Manager nodes (see the example output below).



#### Note

Ensure the pool is created before applying the following specification.

```
service_type: iscsi
service_id: EXAMPLE_ISCSI
placement:
  hosts:
```

```

- ses-min1
- ses-min2
- ses-min3
spec:
  pool: EXAMPLE_POOL
  api_user: EXAMPLE_USER
  api_password: EXAMPLE_PASSWORD
  trusted_ip_list: "IP_ADDRESS_1,IP_ADDRESS_2"

```



## Note

Ensure the IPs listed for `trusted_ip_list` do *not* have a space after the comma separation.

### 8.3.5.1 Secure SSL configuration

To use a secure SSL connection between the Ceph Dashboard and the iSCSI target API, you need a pair of valid SSL certificate and key files. These can be either CA-issued or self-signed (see *Book "Administration and Operations Guide", Chapter 10 "Manual configuration", Section 10.1.1 "Creating self-signed certificates"*). To enable SSL, include the `api_secure: true` setting in your specification file:

```

spec:
  api_secure: true

```

To specify the SSL certificate and key, you can paste the content directly into the YAML specification file. The pipe sign (`|`) at the end of line tells the parser to expect a multi-line string as a value. For example:

```

spec:
  pool: EXAMPLE_POOL
  api_user: EXAMPLE_USER
  api_password: EXAMPLE_PASSWORD
  trusted_ip_list: "IP_ADDRESS_1,IP_ADDRESS_2"
  api_secure: true
  ssl_cert: |
    -----BEGIN CERTIFICATE-----
    MIIDtTCCAp2gAwIBAgIYMC4xNzc1NDQxNjEzMzc2MjMyXzxxvQ7EcMA0GCSqGSIb3
    DQEBChUAMG0xCzAJBgNVBAYTAlVTMQ0wCwYDVQQIDARVdGFoMRcwFQYDVQQHDA5T
    [...]
    -----END CERTIFICATE-----
  ssl_key: |

```

```
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBAcwggSjAgEAAoIBAQC5jdYbjtNTAKW4
/CwQr/7w0iLGzVxChn3mmCIF3DwbL/qvTFTX2d8bDf6LjGwLYLoXHscRfXszX/4h
[...]
-----END PRIVATE KEY-----
```

### 8.3.6 Deploying NFS Ganesha

#### Important

NFS Ganesha supports NFS version 4.1 and newer. It does not support NFS version 3.

cephadm deploys NFS Ganesha using a pre-defined RADOS pool and an optional name-space. To deploy NFS Ganesha, apply the following specification:

#### Note

You need to have a pre-defined RADOS pool otherwise the **ceph orch apply** operation will fail. For more information on creating a pool, see *Book “Administration and Operations Guide”, Chapter 18 “Manage storage pools”, Section 18.1 “Creating a pool”*.

```
service_type: nfs
service_id: EXAMPLE_NFS
placement:
  hosts:
    - ses-min1
    - ses-min2
spec:
  pool: EXAMPLE_POOL
  namespace: EXAMPLE_NAMESPACE
```

- EXAMPLE\_NFS with an arbitrary string that identifies the NFS export.
- EXAMPLE\_POOL with the name of the pool where the NFS Ganesha RADOS configuration object will be stored.
- EXAMPLE\_NAMESPACE (optional) with the desired Object Gateway NFS namespace (for example, ganesha).

### 8.3.7 Deploying rbd-mirror

The `rbd-mirror` service takes care of synchronizing RADOS Block Device images between two Ceph clusters (for more details, see *Book “Administration and Operations Guide”, Chapter 20 “RADOS Block Device”, Section 20.4 “RBD image mirrors”*). To deploy `rbd-mirror`, use the following specification:

```
service_type: rbd-mirror
service_id: EXAMPLE_RBD_MIRROR
placement:
  hosts:
    - ses-min3
```

### 8.3.8 Deploying the monitoring stack

The monitoring stack consists of Prometheus, Prometheus exporters, Prometheus Alertmanager, and Grafana. Ceph Dashboard makes use of these components to store and visualize detailed metrics on cluster usage and performance.



#### Tip

If your deployment requires custom or locally served container images of the monitoring stack services, refer to *Book “Administration and Operations Guide”, Chapter 16 “Monitoring and alerting”, Section 16.1 “Configuring custom or local images”*.

To deploy the monitoring stack, follow these steps:

1. Enable the `prometheus` module in the Ceph Manager daemon. This exposes the internal Ceph metrics so that Prometheus can read them:

```
cephuser@adm > ceph mgr module enable prometheus
```



## Note

Ensure this command is run before Prometheus is deployed. If the command was not run before the deployment, you must redeploy Prometheus to update Prometheus' configuration:

```
cephuser@adm > ceph orch redeploy prometheus
```

2. Create a specification file (for example `monitoring.yaml`) with a content similar to the following:

```
service_type: prometheus
placement:
  hosts:
    - ses-min2
  ---
service_type: node-exporter
  ---
service_type: alertmanager
placement:
  hosts:
    - ses-min4
  ---
service_type: grafana
placement:
  hosts:
    - ses-min3
```

3. Apply monitoring services by running:

```
cephuser@adm > ceph orch apply -i monitoring.yaml
```

It may take a minute or two for the monitoring services to be deployed.



## Important

Prometheus, Grafana, and the Ceph Dashboard are all automatically configured to talk to each other, resulting in a fully functional Grafana integration in the Ceph Dashboard when deployed as described above.

The only exception to this rule is monitoring with RBD images. See *Book "Administration and Operations Guide", Chapter 16 "Monitoring and alerting", Section 16.5.4 "Enabling RBD-image monitoring"* for more information.

## 9 Deployment of additional services

### 9.1 Installation of iSCSI gateway

iSCSI is a storage area network (SAN) protocol that allows clients (called *initiators*) to send SCSI commands to SCSI storage devices (*targets*) on remote servers. SUSE Enterprise Storage 7 includes a facility that opens Ceph storage management to heterogeneous clients, such as Microsoft Windows\* and VMware\* vSphere, through the iSCSI protocol. Multipath iSCSI access enables availability and scalability for these clients, and the standardized iSCSI protocol also provides an additional layer of security isolation between clients and the SUSE Enterprise Storage 7 cluster. The configuration facility is named `ceph-iscsi`. Using `ceph-iscsi`, Ceph storage administrators can define thin-provisioned, replicated, highly-available volumes supporting read-only snapshots, read-write clones, and automatic resizing with Ceph RADOS Block Device (RBD). Administrators can then export volumes either via a single `ceph-iscsi` gateway host, or via multiple gateway hosts supporting multipath failover. Linux, Microsoft Windows, and VMware hosts can connect to volumes using the iSCSI protocol, which makes them available like any other SCSI block device. This means SUSE Enterprise Storage 7 customers can effectively run a complete block-storage infrastructure subsystem on Ceph that provides all the features and benefits of a conventional SAN, enabling future growth.

This chapter introduces detailed information to set up a Ceph cluster infrastructure together with an iSCSI gateway so that the client hosts can use remotely stored data as local storage devices using the iSCSI protocol.

#### 9.1.1 iSCSI block storage

iSCSI is an implementation of the Small Computer System Interface (SCSI) command set using the Internet Protocol (IP), specified in RFC 3720. iSCSI is implemented as a service where a client (the initiator) talks to a server (the target) via a session on TCP port 3260. An iSCSI target's IP address and port are called an *iSCSI portal*, where a target can be exposed through one or more portals. The combination of a target and one or more portals is called the *target portal group* (TPG).

The underlying data link layer protocol for iSCSI is most often Ethernet. More specifically, modern iSCSI infrastructures use 10 GigE Ethernet or faster networks for optimal throughput. 10 Gigabit Ethernet connectivity between the iSCSI gateway and the back-end Ceph cluster is strongly recommended.

#### 9.1.1.1 The Linux kernel iSCSI target

The Linux kernel iSCSI target was originally named LIO for [linux-iscsi.org](http://linux-iscsi.org), the project's original domain and Web site. For some time, no fewer than four competing iSCSI target implementations were available for the Linux platform, but LIO ultimately prevailed as the single iSCSI reference target. The mainline kernel code for LIO uses the simple, but somewhat ambiguous name "target", distinguishing between "target core" and a variety of front-end and back-end target modules.

The most commonly used front-end module is arguably iSCSI. However, LIO also supports Fibre Channel (FC), Fibre Channel over Ethernet (FCoE) and several other front-end protocols. At this time, only the iSCSI protocol is supported by SUSE Enterprise Storage.

The most frequently used target back-end module is one that is capable of simply re-exporting any available block device on the target host. This module is named *iblock*. However, LIO also has an RBD-specific back-end module supporting parallelized multipath I/O access to RBD images.

#### 9.1.1.2 iSCSI initiators

This section introduces brief information on iSCSI initiators used on Linux, Microsoft Windows, and VMware platforms.

##### 9.1.1.2.1 Linux

The standard initiator for the Linux platform is [open-iscsi](http://open-iscsi.org). [open-iscsi](http://open-iscsi.org) launches a daemon, [iscsid](http://iscsid.org), which the user can then use to discover iSCSI targets on any given portal, log in to targets, and map iSCSI volumes. [iscsid](http://iscsid.org) communicates with the SCSI mid layer to create in-kernel block devices that the kernel can then treat like any other SCSI block device on the system. The [open-iscsi](http://open-iscsi.org) initiator can be deployed in conjunction with the Device Mapper Multipath ([dm-multipath](http://dm-multipath.org)) facility to provide a highly available iSCSI block device.

### 9.1.1.2.2 Microsoft Windows and Hyper-V

The default iSCSI initiator for the Microsoft Windows operating system is the Microsoft iSCSI initiator. The iSCSI service can be configured via a graphical user interface (GUI), and supports multipath I/O for high availability.

### 9.1.1.2.3 VMware

The default iSCSI initiator for VMware vSphere and ESX is the VMware ESX software iSCSI initiator, `vmkiscsi`. When enabled, it can be configured either from the vSphere client, or using the `vmkiscsi-tool` command. You can then format storage volumes connected through the vSphere iSCSI storage adapter with VMFS, and use them like any other VM storage device. The VMware initiator also supports multipath I/O for high availability.

## 9.1.2 General information about `ceph-iscsi`

`ceph-iscsi` combines the benefits of RADOS Block Devices with the ubiquitous versatility of iSCSI. By employing `ceph-iscsi` on an iSCSI target host (known as the iSCSI Gateway), any application that needs to make use of block storage can benefit from Ceph, even if it does not speak any Ceph client protocol. Instead, users can use iSCSI or any other target front-end protocol to connect to an LIO target, which translates all target I/O to RBD storage operations.

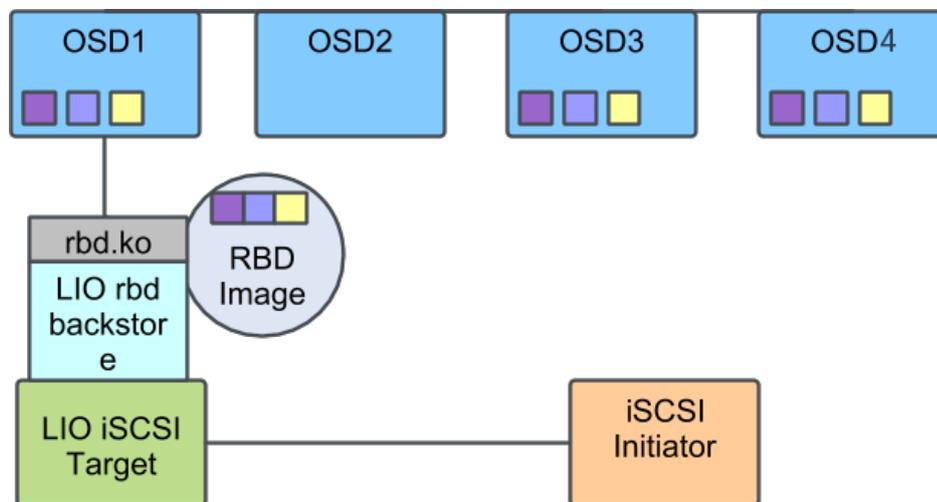


FIGURE 9.1: CEPH CLUSTER WITH A SINGLE ISCSI GATEWAY

`ceph-iscsi` is inherently highly-available and supports multipath operations. Thus, downstream initiator hosts can use multiple iSCSI gateways for both high availability and scalability. When communicating with an iSCSI configuration with more than one gateway, initiators may

load-balance iSCSI requests across multiple gateways. In the event of a gateway failing, being temporarily unreachable, or being disabled for maintenance, I/O will transparently continue via another gateway.

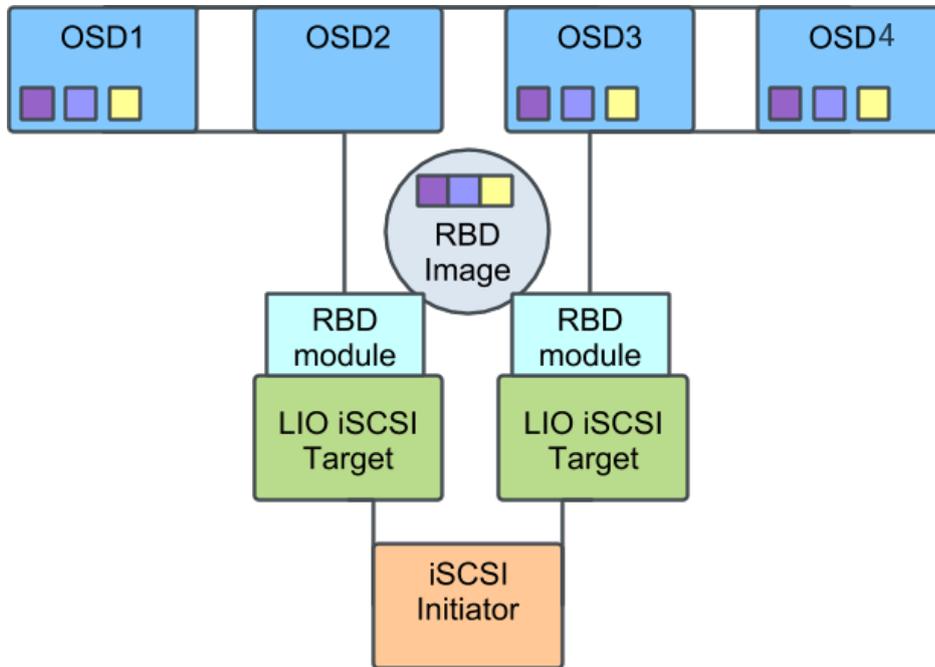


FIGURE 9.2: CEPH CLUSTER WITH MULTIPLE ISCSI GATEWAYS

### 9.1.3 Deployment considerations

A minimum configuration of SUSE Enterprise Storage 7 with `ceph-iscsi` consists of the following components:

- A Ceph storage cluster. The Ceph cluster consists of a minimum of four physical servers hosting at least eight object storage daemons (OSDs) each. In such a configuration, three OSD nodes also double as a monitor (MON) host.
- An iSCSI target server running the LIO iSCSI target, configured via `ceph-iscsi`.
- An iSCSI initiator host, running `open-iscsi` (Linux), the Microsoft iSCSI Initiator (Microsoft Windows), or any other compatible iSCSI initiator implementation.

A recommended production configuration of SUSE Enterprise Storage 7 with `ceph-iscsi` consists of:

- A Ceph storage cluster. A production Ceph cluster consists of any number of (typically more than 10) OSD nodes, each typically running 10-12 object storage daemons (OSDs), with a minimum of three dedicated MON hosts.
- Several iSCSI target servers running the LIO iSCSI target, configured via `ceph-iscsi`. For iSCSI failover and load-balancing, these servers must run a kernel supporting the `target_core_rbd` module. Update packages are available from the SUSE Linux Enterprise Server maintenance channel.
- Any number of iSCSI initiator hosts, running `open-iscsi` (Linux), the Microsoft iSCSI Initiator (Microsoft Windows), or any other compatible iSCSI initiator implementation.

## 9.1.4 Installation and configuration

This section describes steps to install and configure an iSCSI Gateway on top of SUSE Enterprise Storage.

### 9.1.4.1 Deploy the iSCSI Gateway to a Ceph cluster

The Ceph iSCSI Gateway deployment follows the same procedure as the deployment of other Ceph services—by means of `cephadm`. For more details, see [Section 8.3.5, “Deploying iSCSI Gateways”](#).

### 9.1.4.2 Creating RBD images

RBD images are created in the Ceph store and subsequently exported to iSCSI. We recommend that you use a dedicated RADOS pool for this purpose. You can create a volume from any host that is able to connect to your storage cluster using the Ceph `rbd` command line utility. This requires the client to have at least a minimal `ceph.conf` configuration file, and appropriate CephX authentication credentials.

To create a new volume for subsequent export via iSCSI, use the `rbd create` command, specifying the volume size in megabytes. For example, in order to create a 100 GB volume named `testvol` in the pool named `iscsi-images`, run:

```
cephuser@adm > rbd --pool iscsi-images create --size=102400 testvol
```

### 9.1.4.3 Exporting RBD images via iSCSI

To export RBD images via iSCSI, you can use either Ceph Dashboard Web interface or the `ceph-iscsi gwcli` utility. In this section, we will focus on `gwcli` only, demonstrating how to create an iSCSI target that exports an RBD image using the command line.



#### Note

RBD images with the following properties cannot be exported via iSCSI:

- images with the `journaling` feature enabled
- images with a `stripe unit` less than 4096 bytes

As `root`, enter the iSCSI Gateway container:

```
# cephadm enter --name CONTAINER_NAME
```

As `root`, start the iSCSI Gateway command line interface:

```
# gwcli
```

Go to `iscsi-targets` and create a target with the name `iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol`:

```
gwcli > /> cd /iscsi-targets
gwcli > /iscsi-targets> create iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol
```

Create the iSCSI gateways by specifying the gateway `name` and `ip` address:

```
gwcli > /iscsi-targets> cd iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol/
gateways
gwcli > /iscsi-target...tvol/gateways> create iscsi1 192.168.124.104
```

```
gwcli > /iscsi-target...tvol/gateways> create iscsi2 192.168.124.105
```



## Tip

Use the `help` command to show the list of available commands in the current configuration node.

Add the RBD image with the name `testvol` in the pool `iscsi-images` ::

```
gwcli > /iscsi-target...tvol/gateways> cd /disks
gwcli > /disks> attach iscsi-images/testvol
```

Map the RBD image to the target:

```
gwcli > /disks> cd /iscsi-targets/iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol/
disks
gwcli > /iscsi-target...testvol/disks> add iscsi-images/testvol
```



## Note

You can use lower-level tools, such as `targetcli`, to query the local configuration, but not to modify it.



## Tip

You can use the `ls` command to review the configuration. Some configuration nodes also support the `info` command, which can be used to display more detailed information.

Note that, by default, ACL authentication is enabled so this target is not accessible yet. Check [Section 9.1.4.4, "Authentication and access control"](#) for more information about authentication and access control.

### 9.1.4.4 Authentication and access control

iSCSI authentication is flexible and covers many authentication possibilities.

#### 9.1.4.4.1 Disabling ACL authentication

*No Authentication* means that any initiator will be able to access any LUNs on the corresponding target. You can enable *No Authentication* by disabling the ACL authentication:

```
gwcli > /> cd /iscsi-targets/iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol/hosts
gwcli > /iscsi-target...testvol/hosts> auth disable_acl
```

#### 9.1.4.4.2 Using ACL authentication

When using initiator-name-based ACL authentication, only the defined initiators are allowed to connect. You can define an initiator by doing:

```
gwcli > /> cd /iscsi-targets/iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol/hosts
gwcli > /iscsi-target...testvol/hosts> create iqn.1996-04.de.suse:01:e6ca28cc9f20
```

Defined initiators will be able to connect, but will only have access to the RBD images that were explicitly added to the initiator:

```
gwcli > /iscsi-target...:e6ca28cc9f20> disk add rbd/testvol
```

#### 9.1.4.4.3 Enabling CHAP authentication

In addition to the ACL, you can enable CHAP authentication by specifying a user name and password for each initiator:

```
gwcli > /> cd /iscsi-targets/iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol/
hosts/iqn.1996-04.de.suse:01:e6ca28cc9f20
gwcli > /iscsi-target...:e6ca28cc9f20> auth username=common12 password=pass12345678
```



#### Note

User names must have a length of 8 to 64 characters and can contain alphanumeric characters, `.`, `@`, `-`, `_` or `:`.

Passwords must have a length of 12 to 16 characters and can contain alphanumeric characters, `@`, `-`, `_` or `/`.

Optionally, you can also enable CHAP mutual authentication by specifying the `mutual_username` and `mutual_password` parameters in the `auth` command.

#### 9.1.4.4.4 Configuring discovery and mutual authentication

*Discovery authentication* is independent of the previous authentication methods. It requires credentials for browsing, it is optional, and can be configured by:

```
gwcli > /> cd /iscsi-targets
gwcli > /iscsi-targets> discovery_auth username=du123456 password=dp1234567890
```



#### Note

User names must have a length of 8 to 64 characters and can only contain letters, `.`, `@`, `-`, `_` or `:`.

Passwords must have a length of 12 to 16 characters and can only contain letters, `@`, `-`, `_` or `/`.

Optionally, you can also specify the `mutual_username` and `mutual_password` parameters in the `discovery_auth` command.

Discovery authentication can be disabled by using the following command:

```
gwcli > /iscsi-targets> discovery_auth nochap
```

#### 9.1.4.5 Configuring advanced settings

`ceph-iscsi` can be configured with advanced parameters which are subsequently passed on to the LIO I/O target. The parameters are divided up into `target` and `disk` parameters.



#### Warning

Unless otherwise noted, changing these parameters from the default setting is not recommended.

##### 9.1.4.5.1 Viewing target settings

You can view the value of these settings by using the `info` command:

```
gwcli > /> cd /iscsi-targets/iqn.2003-01.org.linux-iscsi.iscsi.SYSTEM-ARCH:testvol
gwcli > /iscsi-target...i.SYSTEM-ARCH:testvol> info
```

And change a setting using the **reconfigure** command:

```
gwcli > /iscsi-target...i.SYSTEM-ARCH:testvol> reconfigure login_timeout 20
```

The available **target** settings are:

#### default\_cmdsn\_depth

Default CmdSN (Command Sequence Number) depth. Limits the amount of requests that an iSCSI initiator can have outstanding at any moment.

#### default\_eri

Default error recovery level.

#### login\_timeout

Login timeout value in seconds.

#### netif\_timeout

NIC failure timeout in seconds.

#### prod\_mode\_write\_protect

If set to 1, prevents writes to LUNs.

### 9.1.4.5.2 Viewing disk settings

You can view the value of these settings by using the **info** command:

```
gwcli > /> cd /disks/rbd/testvol
gwcli > /disks/rbd/testvol> info
```

And change a setting using the **reconfigure** command:

```
gwcli > /disks/rbd/testvol> reconfigure rbd/testvol emulate_pr 0
```

The available **disk** settings are:

#### block\_size

Block size of the underlying device.

#### emulate\_3pc

If set to 1, enables Third Party Copy.

#### emulate\_caw

If set to 1, enables Compare and Write.

#### **emulate\_dpo**

If set to 1, turns on Disable Page Out.

#### **emulate\_fua\_read**

If set to 1, enables Force Unit Access read.

#### **emulate\_fua\_write**

If set to 1, enables Force Unit Access write.

#### **emulate\_model\_alias**

If set to 1, uses the back-end device name for the model alias.

#### **emulate\_pr**

If set to 0, support for SCSI Reservations, including Persistent Group Reservations, is disabled. While disabled, the SES iSCSI Gateway can ignore reservation state, resulting in improved request latency.



#### **Tip**

Setting `backstore_emulate_pr` to 0 is recommended if iSCSI initiators do not require SCSI Reservation support.

#### **emulate\_rest\_reord**

If set to 0, the Queue Algorithm Modifier has Restricted Reordering.

#### **emulate\_tas**

If set to 1, enables Task Aborted Status.

#### **emulate\_tpu**

If set to 1, enables Thin Provisioning Unmap.

#### **emulate\_tpws**

If set to 1, enables Thin Provisioning Write Same.

#### **emulate\_ua\_intlck\_ctrl**

If set to 1, enables Unit Attention Interlock.

#### **emulate\_write\_cache**

If set to 1, turns on Write Cache Enable.

#### **enforce\_pr\_isids**

If set to 1, enforces persistent reservation ISIDs.

#### `is_nonrot`

If set to `1`, the backstore is a non-rotational device.

#### `max_unmap_block_desc_count`

Maximum number of block descriptors for UNMAP.

#### `max_unmap_lba_count:`

Maximum number of LBAs for UNMAP.

#### `max_write_same_len`

Maximum length for WRITE\_SAME.

#### `optimal_sectors`

Optimal request size in sectors.

#### `pi_prot_type`

DIF protection type.

#### `queue_depth`

Queue depth.

#### `unmap_granularity`

UNMAP granularity.

#### `unmap_granularity_alignment`

UNMAP granularity alignment.

#### `force_pr_aptpl`

When enabled, LIO will always write out the *persistent reservation* state to persistent storage, regardless of whether the client has requested it via `aptpl=1`. This has no effect with the kernel RBD back-end for LIO—it always persists PR state. Ideally, the `target_core_rbd` option should force it to '1' and throw an error if someone tries to disable it via configuration.

#### `unmap_zeroes_data`

Affects whether LIO will advertise LBPRZ to SCSI initiators, indicating that zeros will be read back from a region following UNMAP or WRITE SAME with an unmap bit.

### 9.1.5 Exporting RADOS Block Device images using `tcmu-runner`

The `ceph-iscsi` supports both `rbd` (kernel-based) and `user:rbd` (`tcmu-runner`) backstores, making all the management transparent and independent of the backstore.



## Warning: Technology preview

tcmu-runner based iSCSI Gateway deployments are currently a technology preview.

Unlike kernel-based iSCSI Gateway deployments, tcmu-runner based iSCSI Gateways do not offer support for multipath I/O or SCSI Persistent Reservations.

To export an RADOS Block Device image using tcmu-runner, all you need to do is specify the user:rbd backstore when attaching the disk:

```
gwcli > /disks> attach rbd/testvol backstore=user:rbd
```



## Note

When using tcmu-runner, the exported RBD image must have the exclusive-lock feature enabled.

# III Upgrading from Previous Releases

10 Upgrade from a previous release 85

## 10 Upgrade from a previous release



### Important

With the recent update to SUSE Enterprise Storage, it is no longer possible to upgrade from 6 to 7. Customers can now only upgrade from version 6 to version 7.1. Refer to <https://documentation.suse.com/ses/7.1/html/ses-all/ses-upgrade.html> for more details on upgrading to version 7.1.

# A Ceph maintenance updates based on upstream 'Octopus' point releases

Several key packages in SUSE Enterprise Storage 7 are based on the Octopus release series of Ceph. When the Ceph project (<https://github.com/ceph/ceph>) publishes new point releases in the Octopus series, SUSE Enterprise Storage 7 is updated to ensure that the product benefits from the latest upstream bug fixes and feature backports.

This chapter contains summaries of notable changes contained in each upstream point release that has been—or is planned to be—included in the product.

## Octopus 15.2.11 Point Release

This release includes a security fix that ensures the `global_id` value (a numeric value that should be unique for every authenticated client or daemon in the cluster) is reclaimed after a network disconnect or ticket renewal in a secure fashion. Two new health alerts may appear during the upgrade indicating that there are clients or daemons that are not yet patched with the appropriate fix.

To temporarily mute the health alerts around insecure clients for the duration of the upgrade, you may want to run:

```
cephuser@adm > ceph health mute AUTH_INSECURE_GLOBAL_ID_RECLAIM 1h
cephuser@adm > ceph health mute AUTH_INSECURE_GLOBAL_ID_RECLAIM_ALLOWED 1h
```

When all clients are updated, enable the new secure behavior, not allowing old insecure clients to join the cluster:

```
cephuser@adm > ceph config set mon auth_allow_insecure_global_id_reclaim false
```

For more details, refer to <https://docs.ceph.com/en/latest/security/CVE-2021-20288/>.

## Octopus 15.2.10 Point Release

This backport release includes the following fixes:

- The containers include an updated `tcmmalloc` that avoids crashes seen on 15.2.9.
- RADOS: BlueStore handling of huge (> 4GB) writes from RocksDB to BlueFS has been fixed.
- When upgrading from a previous `cephadm` release, `systemctl` may hang when trying to start or restart the monitoring containers. This is caused by a change in the `systemd` unit to use `type=forking`.) After the upgrade, please run:

```
cephuser@adm > ceph orch redeploy nfs
cephuser@adm > ceph orch redeploy iscsi
cephuser@adm > ceph orch redeploy node-exporter
cephuser@adm > ceph orch redeploy prometheus
cephuser@adm > ceph orch redeploy grafana
cephuser@adm > ceph orch redeploy alertmanager
```

## Octopus 15.2.9 Point Release

This backport release includes the following fixes:

- MGR: progress module can now be turned on/off, using the commands: `ceph progress on` and `ceph progress off`.
- OSD: PG removal has been optimized in this release.

## Octopus 15.2.8 Point Release

This release fixes a security flaw in CephFS and includes a number of bug fixes:

- OpenStack Manila use of `ceph_volume_client.py` library allowed tenant access to any Ceph credential's secret.
- **ceph-volume**: The `lvm batch` subcommand received a major rewrite. This closed a number of bugs and improves usability in terms of size specification and calculation, as well as idempotency behaviour and disk replacement process. Please refer to <https://docs.ceph.com/en/latest/ceph-volume/lvm/batch/> for more detailed information.

- **MON:** The cluster log now logs health detail every `mon_health_to_clog_interval`, which has been changed from 1hr to 10min. Logging of health detail will be skipped if there is no change in health summary since last known.
- The `ceph df` command now lists the number of PGs in each pool.
- The `bluefs_preextend_wal_files` option has been removed.
- It is now possible to specify the initial monitor to contact for Ceph tools and daemons using the `mon_host_override` config option or `--mon-host-override` command line switch. This generally should only be used for debugging and only affects initial communication with Ceph's monitor cluster.

## Octopus 15.2.7 Point Release

This release fixes a serious bug in RGW that has been shown to cause data loss when a read of a large RGW object (for example, one with at least one tail segment) takes longer than one half the time specified in the configuration option `rgw_gc_obj_min_wait`. The bug causes the tail segments of that read object to be added to the RGW garbage collection queue, which will in turn cause them to be deleted after a period of time.

## Octopus 15.2.6 Point Release

This releases fixes a security flaw affecting Messenger V2 for Octopus and Nautilus.

## Octopus 15.2.5 Point Release

The Octopus point release 15.2.5 brought the following fixes and other changes:

- **CephFS:** Automatic static sub-tree partitioning policies may now be configured using the new distributed and random ephemeral pinning extended attributes on directories. See the following documentation for more information: <https://docs.ceph.com/docs/master/cephfs/multimds/>
- Monitors now have a configuration option `mon_osd_warn_num_repaired`, which is set to 10 by default. If any OSD has repaired more than this many I/O errors in stored data a `OSD_T00_MANY_REPAIRS` health warning is generated.

- Now, when `no scrub` and/or `no deep-scrub` flags are set globally or per pool, scheduled scrubs of the type disabled will be aborted. All user initiated scrubs are NOT interrupted.
- Fixed an issue with `osdmaps` not being trimmed in a healthy cluster.

## Octopus 15.2.4 Point Release

The Octopus point release 15.2.4 brought the following fixes and other changes:

- CVE-2020-10753: `rgw`: sanitize newlines in `s3 CORSConfiguration`'s `ExposeHeader`
- Object Gateway: The `radosgw-admin` sub-commands dealing with orphans—`radosgw-admin orphans find`, `radosgw-admin orphans finish`, and `radosgw-admin orphans list-jobs`—have been deprecated. They had not been actively maintained, and since they store intermediate results on the cluster, they could potentially fill a nearly-full cluster. They have been replaced by a tool, `rgw-orphan-list`, which is currently considered experimental.
- RBD: The name of the RBD pool object that is used to store RBD trash purge schedule is changed from `rbd_trash_trash_purge_schedule` to `rbd_trash_purge_schedule`. Users that have already started using RBD trash purge schedule functionality and have per pool or name space schedules configured should copy the `rbd_trash_trash_purge_schedule` object to `rbd_trash_purge_schedule` before the upgrade and remove `rbd_trash_purge_schedule` using the following commands in every RBD pool and name space where a trash purge schedule was previously configured:

```
rados -p pool-name [-N namespace] cp rbd_trash_trash_purge_schedule
  rbd_trash_purge_schedule
rados -p pool-name [-N namespace] rm rbd_trash_trash_purge_schedule
```

Alternatively, use any other convenient way to restore the schedule after the upgrade.

## Octopus 15.2.3 Point Release

- The Octopus point release 15.2.3 was a hot-fix release to address an issue where WAL corruption was seen when `bluefs_preextend_wal_files` and `bluefs_buffered_io` were enabled at the same time. The fix in 15.2.3 is only a temporary measure (changing the

default value of `bluefs_preextend_wal_files` to `false`). The permanent fix will be to remove the `bluefs_preextend_wal_files` option completely: this fix will most likely arrive in the 15.2.6 point release.

## Octopus 15.2.2 Point Release

The Octopus point release 15.2.2 patched one security vulnerability:

- CVE-2020-10736: Fixed an authorization bypass in MONs and MGRs

## Octopus 15.2.1 Point Release

The Octopus point release 15.2.1 fixed an issue where upgrading quickly from Luminous (SES5.5) to Nautilus (SES6) to Octopus (SES7) caused OSDs to crash. In addition, it patched two security vulnerabilities that were present in the initial Octopus (15.2.0) release:

- CVE-2020-1759: Fixed nonce reuse in msgr V2 secure mode
- CVE-2020-1760: Fixed XSS because of RGW GetObject header-splitting

# Glossary

## General

### **Admin node**

The host from which you run the Ceph-related commands to administer cluster hosts.

### **Alertmanager**

A single binary which handles alerts sent by the Prometheus server and notifies the end user.

### **archive sync module**

Module that enables creating an Object Gateway zone for keeping the history of S3 object versions.

### **Bucket**

A point that aggregates other nodes into a hierarchy of physical locations.

### **Ceph Client**

The collection of Ceph components which can access a Ceph Storage Cluster. These include the Object Gateway, the Ceph Block Device, the CephFS, and their corresponding libraries, kernel modules, and FUSE clients.

### **Ceph Dashboard**

A built-in Web-based Ceph management and monitoring application to administer various aspects and objects of the cluster. The dashboard is implemented as a Ceph Manager module.

### **Ceph Manager**

Ceph Manager or MGR is the Ceph manager software, which collects all the state from the whole cluster in one place.

### **Ceph Monitor**

Ceph Monitor or MON is the Ceph monitor software.

### **Ceph Object Storage**

The object storage "product", service or capabilities, which consists of a Ceph Storage Cluster and a Ceph Object Gateway.

## Ceph OSD Daemon

The **ceph-osd** daemon is the component of Ceph that is responsible for storing objects on a local file system and providing access to them over the network.

## Ceph Storage Cluster

The core set of storage software which stores the user's data. Such a set consists of Ceph monitors and OSDs.

### ceph-salt

Provides tooling for deploying Ceph clusters managed by cephadm using Salt.

## cephadm

cephadm deploys and manages a Ceph cluster by connecting to hosts from the manager daemon via SSH to add, remove, or update Ceph daemon containers.

## CephFS

The Ceph file system.

## CephX

The Ceph authentication protocol. Cephx operates like Kerberos, but it has no single point of failure.

## CRUSH rule

The CRUSH data placement rule that applies to a particular pool or pools.

## CRUSH, CRUSH Map

*Controlled Replication Under Scalable Hashing*: An algorithm that determines how to store and retrieve data by computing data storage locations. CRUSH requires a map of the cluster to pseudo-randomly store and retrieve data in OSDs with a uniform distribution of data across the cluster.

## DriveGroups

DriveGroups are a declaration of one or more OSD layouts that can be mapped to physical drives. An OSD layout defines how Ceph physically allocates OSD storage on the media matching the specified criteria.

## Grafana

Database analytics and monitoring solution.

## Metadata Server

Metadata Server or MDS is the Ceph metadata software.

## Multi-zone

## Node

Any single machine or server in a Ceph cluster.

## Object Gateway

The S3/Swift gateway component for Ceph Object Store. Also known as the RADOS Gateway (RGW).

## OSD

*Object Storage Device*: A physical or logical storage unit.

## OSD node

A cluster node that stores data, handles data replication, recovery, backfilling, rebalancing, and provides some monitoring information to Ceph monitors by checking other Ceph OSD daemons.

## PG

Placement Group: a sub-division of a *pool*, used for performance tuning.

## Point Release

Any ad-hoc release that includes only bug or security fixes.

## Pool

Logical partitions for storing objects such as disk images.

## Prometheus

Systems monitoring and alerting toolkit.

## RADOS Block Device (RBD)

The block storage component of Ceph. Also known as the Ceph block device.

## Reliable Autonomic Distributed Object Store (RADOS)

The core set of storage software which stores the user's data (MON + OSD).

## Routing tree

A term given to any diagram that shows the various routes a receiver can run.

## Rule Set

Rules to determine data placement for a pool.

## **Samba**

Windows integration software.

## **Samba Gateway**

The Samba Gateway joins the Active Directory in the Windows domain to authenticate and authorize users.

## **zonegroup**