

# Deploy Hyper-Converged Ceph Cluster

---

**Proxmox Server Solutions GmbH**

[<support@proxmox.com>](mailto:support@proxmox.com)

version 8.2.2, Thu Apr 25 09:24:16 CEST 2024

[↶ Index](#)

## Table of Contents

Introduction

Terminology

Recommendations for a Healthy Ceph Cluster

Initial Ceph Installation & Configuration

Using the Web-based Wizard

CLI Installation of Ceph Packages

Initial Ceph configuration via CLI

Ceph Monitor

Create Monitors

Destroy Monitors

Ceph Manager

Create Manager

Destroy Manager

Ceph OSDs

Create OSDs

Destroy OSDs

Ceph Pools

Create and Edit Pools

Erasured Pools

Destroy Pools

PG Autoscaler

Ceph CRUSH & device classes

Ceph Client

CephFS

Metadata Server (MDS)

Create CephFS

Destroy CephFS

Ceph maintenance

Replace OSDs

Trim/Discard

Scrub & Deep Scrub

Ceph Monitoring and Troubleshooting

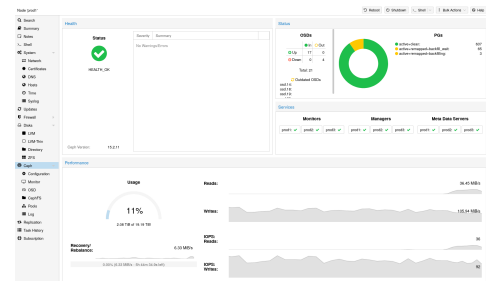
---

## Introduction

Proxmox VE unifies your compute and storage systems, that is, you can use the same physical nodes

within a cluster for both computing (processing VMs and containers) and replicated storage. The traditional silos of compute and storage resources can be wrapped up into a single hyper-converged appliance. Separate storage networks (SANs) and connections via network attached storage (NAS) disappear. With the integration of Ceph, an open source software-defined storage platform, Proxmox VE has the ability to run and manage Ceph storage directly on the hypervisor nodes.

Ceph is a distributed object store and file system designed to provide excellent performance, reliability and scalability.



### Some advantages of Ceph on Proxmox VE are:

- Easy setup and management via CLI and GUI
- Thin provisioning
- Snapshot support
- Self healing
- Scalable to the exabyte level
- Provides block, file system, and object storage
- Setup pools with different performance and redundancy characteristics
- Data is replicated, making it fault tolerant
- Runs on commodity hardware
- No need for hardware RAID controllers
- Open source

For small to medium-sized deployments, it is possible to install a Ceph server for using RADOS Block Devices (RBD) or CephFS directly on your Proxmox VE cluster nodes (see [Ceph RADOS Block Devices \(RBD\)](#)). Recent hardware has a lot of CPU power and RAM, so running storage services and virtual guests on the same node is possible.

To simplify management, Proxmox VE provides you native integration to install and manage [Ceph](#) services on Proxmox VE nodes either via the built-in web interface, or using the *pveceph* command line tool.

## Terminology

## Ceph consists of multiple Daemons, for use as an RBD storage:

- Ceph Monitor (ceph-mon, or MON)
- Ceph Manager (ceph-mgr, or MGS)
- Ceph Metadata Service (ceph-mds, or MDS)
- Ceph Object Storage Daemon (ceph-osd, or OSD)



We highly recommend to get familiar with Ceph [\[1\]](#), its architecture [\[2\]](#) and vocabulary [\[3\]](#).

## Recommendations for a Healthy Ceph Cluster

To build a hyper-converged Proxmox + Ceph Cluster, you must use at least three (preferably) identical servers for the setup.

Check also the recommendations from [Ceph's website](#).



The recommendations below should be seen as a rough guidance for choosing hardware. Therefore, it is still essential to adapt it to your specific needs. You should test your setup and monitor health and performance continuously.

### CPU

Ceph services can be classified into two categories: \* Intensive CPU usage, benefiting from high CPU base frequencies and multiple cores. Members of that category are: **Object Storage Daemon (OSD) services** Meta Data Service (MDS) used for CephFS \* Moderate CPU usage, not needing multiple CPU cores. These are: **Monitor (MON) services** Manager (MGR) services

As a simple rule of thumb, you should assign at least one CPU core (or thread) to each Ceph service to provide the minimum resources required for stable and durable Ceph performance.

For example, if you plan to run a Ceph monitor, a Ceph manager and 6 Ceph OSDs services on a node you should reserve 8 CPU cores purely for Ceph when targeting basic and stable performance.

Note that OSDs CPU usage depend mostly from the disks performance. The higher the possible IOPS (**I**O **O**perations per **S**econd) of a disk, the more CPU can be utilized by a OSD service. For modern

enterprise SSD disks, like NVMe's that can permanently sustain a high IOPS load over 100'000 with sub millisecond latency, each OSD can use multiple CPU threads, e.g., four to six CPU threads utilized per NVMe backed OSD is likely for very high performance disks.

## Memory

Especially in a hyper-converged setup, the memory consumption needs to be carefully planned out and monitored. In addition to the predicted memory usage of virtual machines and containers, you must also account for having enough memory available for Ceph to provide excellent and stable performance.

As a rule of thumb, for roughly **1 TiB of data, 1 GiB of memory** will be used by an OSD. While the usage might be less under normal conditions, it will use most during critical operations like recovery, re-balancing or backfilling. That means that you should avoid maxing out your available memory already on normal operation, but rather leave some headroom to cope with outages.

The OSD service itself will use additional memory. The Ceph BlueStore backend of the daemon requires by default **3-5 GiB of memory** (adjustable).

## Network

We recommend a network bandwidth of at least 10 Gbps, or more, to be used exclusively for Ceph traffic. A meshed network setup <sup>[4]</sup> is also an option for three to five node clusters, if there are no 10+ Gbps switches available.

- The volume of traffic, especially during recovery, will interfere with other services on the same network, especially the latency sensitive Proxmox VE corosync cluster stack can be affected, resulting in possible loss of cluster quorum. Moving the Ceph traffic to dedicated and physical separated networks will avoid such interference, not only for corosync, but also for the networking services provided by any virtual guests.

For estimating your bandwidth needs, you need to take the performance of your disks into account.. While a single HDD might not saturate a 1 Gb link, multiple HDD OSDs per node can already saturate 10 Gbps too. If modern NVMe-attached SSDs are used, a single one can already saturate 10 Gbps of bandwidth, or more. For such high-performance

setups we recommend at least a 25 Gbps, while even 40 Gbps or 100+ Gbps might be required to utilize the full performance potential of the underlying disks.

If unsure, we recommend using three (physical) separate networks for high-performance setups: \* one very high bandwidth (25+ Gbps) network for Ceph (internal) cluster traffic. \* one high bandwidth (10+ Gbps) network for Ceph (public) traffic between the ceph server and ceph client storage traffic. Depending on your needs this can also be used to host the virtual guest traffic and the VM live-migration traffic. \* one medium bandwidth (1 Gbps) exclusive for the latency sensitive corosync cluster communication.

## Disks

When planning the size of your Ceph cluster, it is important to take the recovery time into consideration. Especially with small clusters, recovery might take long. It is recommended that you use SSDs instead of HDDs in small setups to reduce recovery time, minimizing the likelihood of a subsequent failure event during recovery.

In general, SSDs will provide more IOPS than spinning disks. With this in mind, in addition to the higher cost, it may make sense to implement a [class based](#) separation of pools. Another way to speed up OSDs is to use a faster disk as a journal or DB/**W**rite-**A**head-**L**og device, see [creating Ceph OSDs](#). If a faster disk is used for multiple OSDs, a proper balance between OSD and WAL / DB (or journal) disk must be selected, otherwise the faster disk becomes the bottleneck for all linked OSDs.

Aside from the disk type, Ceph performs best with an evenly sized, and an evenly distributed amount of disks per node. For example, 4 x 500 GB disks within each node is better than a mixed setup with a single 1 TB and three 250 GB disk.

You also need to balance OSD count and single OSD capacity. More capacity allows you to increase storage density, but it also means that a single OSD failure forces Ceph to recover more data at once.

## Avoid RAID

As Ceph handles data object redundancy and multiple parallel writes to disks (OSDs) on its own, using a RAID controller normally doesn't improve performance or availability. On the contrary, Ceph is designed to handle whole disks on its own, without any abstraction in between. RAID controllers are not designed for the Ceph workload and may complicate things and sometimes even reduce performance, as their write and caching algorithms may interfere with the ones from Ceph.



Avoid RAID controllers. Use host bus adapter (HBA) instead.

## Initial Ceph Installation & Configuration

### Using the Web-based Wizard

With Proxmox VE you have the benefit of an easy to use installation wizard for Ceph. Click on one of your cluster nodes and navigate to the Ceph

section in the menu tree. If Ceph is not already installed, you will see a prompt offering to do so.

The wizard is divided into multiple sections, where each needs to finish successfully, in order to use Ceph.

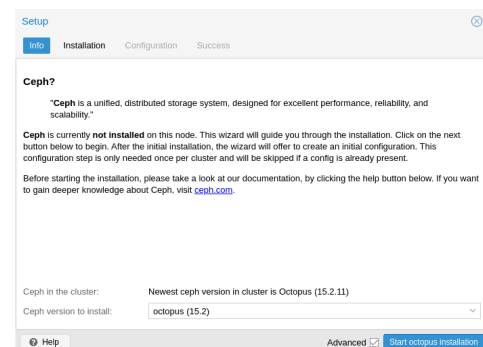
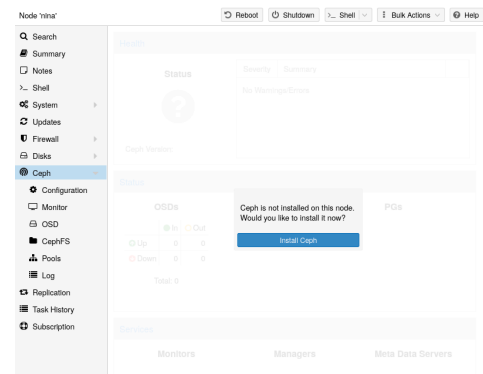
First you need to choose which Ceph version you want to install. Prefer the one from your other nodes, or the newest if this is the first node you install Ceph.

After starting the installation, the wizard will download and install all the required packages from Proxmox VE's Ceph repository.

After finishing the installation step, you will need to create a configuration. This step is only needed once per cluster, as this configuration is distributed automatically to all remaining cluster members through Proxmox VE's clustered [configuration file system \(pmxcfs\)](#).

The configuration step includes the following settings:

- **Public Network:** This network will be used for public storage communication (e.g., for virtual machines using a Ceph RBD backed disk, or a CephFS mount), and communication between the different Ceph



services. This setting is required.

Separating your Ceph traffic from the Proxmox VE cluster communication (corosync), and possible the front-facing (public) networks of your virtual guests, is highly recommended. Otherwise, Ceph's high-bandwidth IO-traffic could cause interference with other low-latency dependent services.

- **Cluster Network:** Specify to separate the [OSD](#) replication and heartbeat traffic as well. This setting is optional.

Using a physically separated network is recommended, as it will relieve the Ceph public and the virtual guests network, while also providing a significant Ceph performance improvements.

The Ceph cluster network can be configured and moved to another physically separated network at a later time.

You have two more options which are considered advanced and therefore should only be changed if you know what you are doing.

- **Number of replicas:** Defines how often an object is replicated.
- **Minimum replicas:** Defines the minimum number of required replicas for I/O to be marked as complete.

Additionally, you need to choose your first monitor node. This step is required.

That's it. You should now see a success page as the last step, with further instructions on how to proceed. Your system is now ready to start using Ceph. To get started, you will need to create some additional [monitors](#), [OSDs](#) and at least one [pool](#).

The rest of this chapter will guide you through getting the most out of your Proxmox VE based Ceph setup. This includes the aforementioned tips and more, such as [CephFS](#), which is a helpful addition to your new Ceph cluster.

## CLI Installation of Ceph Packages

Alternatively to the the recommended Proxmox VE Ceph installation wizard available in the web interface, you can use the following CLI command on each node:

```
pveceph install
```

This sets up an `apt` package repository in `/etc/apt/sources.list.d/ceph.list` and installs the required software.

## Initial Ceph configuration via CLI

Use the Proxmox VE Ceph installation wizard (recommended) or run the following command on one node:

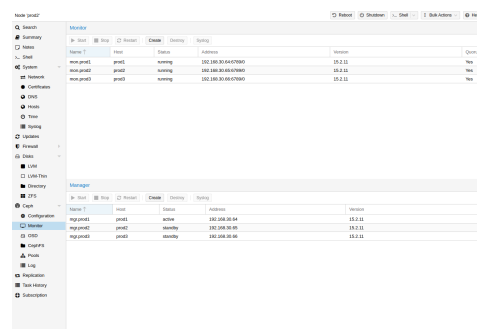
```
pveceph init --network 10.10.10.0/24
```

This creates an initial configuration at `/etc/pve/ceph.conf` with a dedicated network for Ceph. This file is automatically distributed to all Proxmox VE nodes, using `pmxcfs`. The command also creates a symbolic link at `/etc/ceph/ceph.conf`, which points to that file. Thus, you can simply run Ceph commands without the need to specify a configuration file.

## Ceph Monitor

The Ceph Monitor (MON) [5] maintains a master copy of the cluster map. For high availability, you need at

least 3 monitors. One monitor will already be installed if you used the installation wizard. You won't need more than 3 monitors, as long as your cluster is small to medium-sized. Only really large clusters will require more than this.



Monitor	IP	Port	Weight	Status	Quorum
mon0	10.10.10.10	3300	1	OK	Yes
mon1	10.10.10.11	3300	1	OK	Yes
mon2	10.10.10.12	3300	1	OK	Yes

## Create Monitors

On each node where you want to place a monitor (three monitors are recommended), create one by using the *Ceph → Monitor* tab in the GUI or run:

```
pveceph mon create
```

## Destroy Monitors



To remove a Ceph Monitor via the GUI, first select a node in the tree view and go to the **Ceph → Monitor** panel. Select the MON and click the **Destroy** button.

To remove a Ceph Monitor via the CLI, first connect to the node on which the MON is running. Then execute the following command:

```
pveceph mon destroy
```



At least three Monitors are needed for quorum.

## Ceph Manager

The Manager daemon runs alongside the monitors. It provides an interface to monitor the cluster. Since the release of Ceph luminous, at least one ceph-mgr [6] daemon is required.

### Create Manager

Multiple Managers can be installed, but only one Manager is active at any given time.

```
pveceph mgr create
```



It is recommended to install the Ceph Manager on the monitor nodes. For high availability install more than one manager.

### Destroy Manager

To remove a Ceph Manager via the GUI, first select a node in the tree view and go to the **Ceph → Monitor** panel. Select the Manager and click the **Destroy** button.

To remove a Ceph Monitor via the CLI, first connect to the node on which the Manager is running. Then execute the following command:

```
pveceph mgr destroy
```



While a manager is not a hard-dependency, it is crucial for a Ceph cluster, as it handles important features

like PG-autoscaling, device health monitoring, telemetry and more.

## Ceph OSDs

**Ceph Object Storage Daemons** store objects for Ceph over the network. It is recommended to use one OSD per physical disk.

Name	Class	OSD Type	Status	Version	Weight	Size
osd0	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd1	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd2	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd3	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd4	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd5	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd6	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd7	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd8	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd9	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd10	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd11	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd12	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd13	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd14	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd15	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd16	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd17	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd18	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd19	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd20	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd21	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd22	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd23	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd24	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd25	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd26	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd27	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd28	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd29	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd30	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd31	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd32	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd33	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd34	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd35	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd36	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd37	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd38	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd39	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd40	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd41	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd42	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd43	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd44	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd45	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd46	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd47	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd48	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd49	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd50	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd51	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd52	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd53	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd54	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd55	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd56	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd57	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd58	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd59	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd60	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd61	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd62	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd63	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd64	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd65	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd66	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd67	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd68	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd69	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd70	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd71	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd72	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd73	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd74	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd75	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd76	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd77	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd78	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd79	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd80	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd81	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd82	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd83	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd84	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd85	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd86	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd87	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd88	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd89	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd90	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd91	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd92	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd93	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd94	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd95	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd96	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd97	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd98	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd99	bluestore	bluestore	up	16.0.0	1.00	1.00 TB
osd100	bluestore	bluestore	up	16.0.0	1.00	1.00 TB

## Create OSDs

You can create an OSD either via the Proxmox VE web interface or via the CLI using `pveceph`. For example:

```
pveceph osd create /dev/sd[X]
```

**i** We recommend a Ceph cluster with at least three nodes and at least 12 OSDs, evenly distributed among the nodes.

If the disk was in use before (for example, for ZFS or as an OSD) you first need to zap all traces of that usage. To remove the partition table, boot sector and any other OSD leftover, you can use the following command:

```
ceph-volume lvm zap /dev/sd[X] --destroy
```



The above command will destroy all data on the disk!

## Ceph Bluestore

Starting with the Ceph Kraken release, a new Ceph OSD storage type was introduced called Bluestore [2]. This is the default when creating OSDs since Ceph Luminous.

```
pveceph osd create /dev/sd[X]
```

## Block.db and block.wal

If you want to use a separate DB/WAL device for your OSDs, you can specify it through the `-db_dev` and `-wal_dev` options. The WAL is placed with the DB, if not specified separately.

```
pveceph osd create /dev/sd[X] -db_dev /dev/sd[Y]
-wal_dev /dev/sd[Z]
```

You can directly choose the size of those with the `-db_size` and `-wal_size` parameters respectively. If they are not given, the following values (in order) will be used:

- `bluestore_block_{db,wal}_size` from Ceph configuration...
  - ... database, section `osd`
  - ... database, section `global`
  - ... file, section `osd`
  - ... file, section `global`
- 10% (DB)/1% (WAL) of OSD size



The DB stores BlueStore's internal metadata, and the WAL is BlueStore's internal journal or write-ahead log. It is recommended to use a fast SSD or NVRAM for better performance.

## Ceph Filestore

Before Ceph Luminous, Filestore was used as the default storage type for Ceph OSDs. Starting with Ceph Nautilus, Proxmox VE does not support creating such OSDs with `pveceph` anymore. If you still want to create filestore OSDs, use `ceph-volume` directly.

```
ceph-volume lvm create --filestore --data
/dev/sd[X] --journal /dev/sd[Y]
```

## Destroy OSDs

To remove an OSD via the GUI, first select a Proxmox VE node in the tree view and go to the **Ceph → OSD** panel. Then select the OSD to destroy and click the **OUT** button. Once the OSD status has changed from `in` to `out`, click the **STOP** button. Finally, after the status has changed from `up` to `down`, select **Destroy** from the `More` drop-down menu.

To remove an OSD via the CLI run the following commands.

```
ceph osd out <ID>
systemctl stop ceph-osd@<ID>.service
```



The first command instructs Ceph not to include the OSD in the data distribution. The second command stops the OSD service. Until this time, no data is lost.

The following command destroys the OSD. Specify the **-cleanup** option to additionally destroy the partition table.

```
pveceph osd destroy <ID>
```



The above command will destroy all data on the disk!

## Ceph Pools

A pool is a logical group for storing objects. It holds a collection of objects, known as **Placement Groups (PG, pg\_num)**.

Name	Size	# of Placement Gr.	Optimal # of PGs	Autoscaler Mode	CR (Current PGs)	Used PGs
pg1	32	32	32	on	replicated, v0 (0)	807.54 GB (0.00%)
pg2	32	32	32	on	replicated, v0 (0)	808.58 GB (0.00%)
pg3	32	32	32	on	replicated, v0 (0)	811.36 GB (0.00%)
pg4	32	32	32	on	replicated, v0 (0)	813.15 GB (0.00%)
pg5	32	32	32	on	replicated, v0 (0)	814.94 GB (0.00%)
pg6	32	32	32	on	replicated, v0 (0)	816.73 GB (0.00%)

## Create and Edit Pools

You can create and edit pools from the command line or the web interface of any Proxmox VE host under **Ceph → Pools**.

When no options are given, we set a default of **128 PGs**, a **size of 3 replicas** and a **min\_size of 2 replicas**, to ensure no data loss occurs if any OSD fails.



**Do not set a min\_size of 1.** A replicated pool with min\_size of 1 allows I/O on an object when it has only 1 replica, which could lead to data loss, incomplete PGs or unfound objects.

It is advised that you either enable the PG-Autoscaler or calculate the PG number based on

your setup. You can find the formula and the PG calculator [\[8\]](#) online. From Ceph Nautilus onward, you can change the number of PGs [\[9\]](#) after the setup.

The PG autoscaler [\[10\]](#) can automatically scale the PG count for a pool in the background. Setting the `Target Size` or `Target Ratio` advanced parameters helps the PG-Autoscaler to make better decisions.

### Example for creating a pool over the CLI

```
pveceph pool create <pool-name> --add_storages
```

- If you would also like to automatically define a storage for your pool, keep the ‘Add as Storage’ checkbox checked in the web interface, or use the command-line option `--add_storages` at pool creation.

## Pool Options

The following options are available on pool creation, and partially also when editing a pool.

### Name

The name of the pool. This must be unique and can't be changed afterwards.

### Size

The number of replicas per object. Ceph always tries to have this many copies of an object. Default: 3.

### PG Autoscale Mode

The automatic PG scaling mode [\[10\]](#) of the pool. If set to `warn`, it produces a warning message when a pool has a non-optimal PG count. Default: `warn`.

### Add as Storage

Configure a VM or container storage using the new pool. Default: `true` (only visible on creation).

## Advanced Options

### Min. Size

The minimum number of replicas per object. Ceph will reject I/O on the pool if a PG has less than this many replicas. Default: 2.

### Crush Rule

The rule to use for mapping object placement in the cluster. These rules define how data is placed within the cluster. See [Ceph CRUSH & device classes](#) for information on device-based rules.

### # of PGs

The number of placement groups <sup>[9]</sup> that the pool should have at the beginning. Default: 128.

### Target Ratio

The ratio of data that is expected in the pool. The PG autoscaler uses the ratio relative to other ratio sets. It takes precedence over the `target_size` if both are set.

### Target Size

The estimated amount of data expected in the pool. The PG autoscaler uses this size to estimate the optimal PG count.

### Min. # of PGs

The minimum number of placement groups. This setting is used to fine-tune the lower bound of the PG count for that pool. The PG autoscaler will not merge PGs below this threshold.

Further information on Ceph pool handling can be found in the Ceph pool operation <sup>[1]</sup> manual.

## Erasure Coded Pools

Erasure coding (EC) is a form of ‘forward error correction’ codes that allows to recover from a certain amount of data loss. Erasure coded pools can offer more usable space compared to replicated pools, but they do that for the price of performance.

For comparison: in classic, replicated pools, multiple replicas of the data are stored (`size`) while in erasure coded pool, data is split into `k` data chunks with additional `m` coding (checking) chunks. Those coding chunks can be used to recreate data should data chunks be missing.

The number of coding chunks, `m`, defines how many OSDs can be lost without losing any data. The total amount of objects stored is `k + m`.

## Creating EC Pools

Erasure coded (EC) pools can be created with the `pveceph` CLI tooling. Planning an EC pool needs to account for the fact, that they work differently than replicated pools.

The default `min_size` of an EC pool depends on the `m` parameter. If `m = 1`, the `min_size` of the EC

pool will be  $k$ . The `min_size` will be  $k + 1$  if  $m > 1$ . The Ceph documentation recommends a conservative `min_size` of  $k + 2$  [12].

If there are less than `min_size` OSDs available, any IO to the pool will be blocked until there are enough OSDs available again.



When planning an erasure coded pool, keep an eye on the `min_size` as it defines how many OSDs need to be available. Otherwise, IO will be blocked.

For example, an EC pool with  $k = 2$  and  $m = 1$  will have `size = 3`, `min_size = 2` and will stay operational if one OSD fails. If the pool is configured with  $k = 2$ ,  $m = 2$ , it will have a `size = 4` and `min_size = 3` and stay operational if one OSD is lost.

To create a new EC pool, run the following command:

```
pveceph pool create <pool-name> --erasure-coding
k=2,m=1
```

Optional parameters are `failure-domain` and `device-class`. If you need to change any EC profile settings used by the pool, you will have to create a new pool with a new profile.

This will create a new EC pool plus the needed replicated pool to store the RBD omap and other metadata. In the end, there will be a `<pool name>-data` and `<pool name>-metada` pool. The default behavior is to create a matching storage configuration as well. If that behavior is not wanted, you can disable it by providing the `--add_storages 0` parameter. When configuring the storage configuration manually, keep in mind that the `data-pool` parameter needs to be set. Only then will the EC pool be used to store the data objects. For example:



The optional parameters `--size`, `--min_size` and `--crush_rule` will be used for the replicated metadata pool, but not for the erasure coded data pool. If you need to change the `min_size` on the data pool, you can do it later. The `size` and `crush_rule` parameters cannot be changed on erasure coded pools.

If there is a need to further customize the EC profile, you can do so by creating it with the Ceph tools directly [\[13\]](#), and specify the profile to use with the `profile` parameter.

For example:

```
pveceph pool create <pool-name> --erasure-coding  
profile=<profile-name>
```

## Adding EC Pools as Storage

You can add an already existing EC pool as storage to Proxmox VE. It works the same way as adding an RBD pool but requires the extra `data-pool` option.

```
pvesm add rbd <storage-name> --pool <replicated-  
pool> --data-pool <ec-pool>
```

- Do not forget to add the `keyring` and `monhost` option for any external Ceph clusters, not managed by the local Proxmox VE cluster.

## Destroy Pools

To destroy a pool via the GUI, select a node in the tree view and go to the **Ceph → Pools** panel. Select the pool to destroy and click the **Destroy** button. To confirm the destruction of the pool, you need to enter the pool name.

Run the following command to destroy a pool. Specify the `-remove_storages` to also remove the associated storage.

```
pveceph pool destroy <name>
```

- Pool deletion runs in the background and can take some time. You will notice the data usage in the cluster decreasing throughout this process.

## PG Autoscaler

The PG autoscaler allows the cluster to consider the amount of (expected) data stored in each pool and to choose the appropriate `pg_num` values automatically. It is available since Ceph Nautilus.

You may need to activate the PG autoscaler module before adjustments can take effect.



```
ceph mgr module enable pg_autoscaler
```

The autoscaler is configured on a per pool basis and has the following modes:

- warn** A health warning is issued if the suggested `pg_num` value differs too much from the current value.
- on** The `pg_num` is adjusted automatically with no need for any manual interaction.
- off** No automatic `pg_num` adjustments are made, and no warning will be issued if the PG count is not optimal.

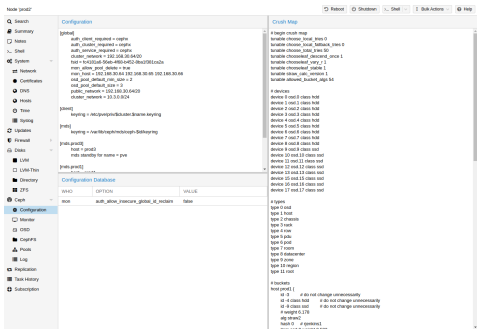
The scaling factor can be adjusted to facilitate future data storage with the `target_size`, `target_size_ratio` and the `pg_num_min` options.

- By default, the autoscaler considers tuning the PG count of a pool if it is off by a factor of 3. This will lead to a considerable shift in data placement and might introduce a high load on the cluster.

You can find a more in-depth introduction to the PG autoscaler on Ceph’s Blog - [New in Nautilus: PG merging and autotuning](#).

# Ceph CRUSH & device classes

The <sup>[14]</sup>  
(Controlled  
Replication  
Under  
Scalable  
Hashing)  
algorithm is at  
the foundation  
of Ceph.



CRUSH calculates where to store and retrieve data from. This has the advantage that no central indexing service is needed. CRUSH works using a map of OSDs, buckets (device locations) and rulesets (data replication) for pools.

- Further information can be found in the Ceph documentation, under the section CRUSH map <sup>[15]</sup>.

This map can be altered to reflect different replication hierarchies. The object replicas can be separated (e.g., failure domains), while maintaining the desired distribution.

A common configuration is to use different classes of disks for different Ceph pools. For this reason, Ceph introduced device classes with luminous, to accommodate the need for easy ruleset generation.

The device classes can be seen in the *ceph osd tree* output. These classes represent their own root bucket, which can be seen with the below command.

```
ceph osd crush tree --show-shadow
```

Example output form the above command:

```
ID CLASS WEIGHT TYPE NAME
-16 nvme 2.18307 root default~nvme
-13 nvme 0.72769 host sumi1~nvme
12 nvme 0.72769 osd.12
-14 nvme 0.72769 host sumi2~nvme
13 nvme 0.72769 osd.13
-15 nvme 0.72769 host sumi3~nvme
14 nvme 0.72769 osd.14
-1 7.70544 root default
-3 2.56848 host sumi1
12 nvme 0.72769 osd.12
-5 2.56848 host sumi2
13 nvme 0.72769 osd.13
-7 2.56848 host sumi3
14 nvme 0.72769 osd.14
```

To instruct a pool to only distribute objects on a specific device class, you first need to create a ruleset for the device class:

```
ceph osd crush rule create-replicated <rule-name> <root> <failure-domain> <class>
```

<rule-name>	name of the rule, to connect with a pool (seen in GUI & CLI)
<root>	which crush root it should belong to (default Ceph root "default")
<failure-domain>	at which failure-domain the objects should be distributed (usually host)
<class>	what type of OSD backing store to use (e.g., nvme, ssd, hdd)

Once the rule is in the CRUSH map, you can tell a pool to use the ruleset.

```
ceph osd pool set <pool-name> crush_rule <rule-name>
```

- If the pool already contains objects, these must be moved accordingly. Depending on your setup, this may introduce a big performance impact on your cluster. As an alternative, you can create a new pool and move disks separately.

## Ceph Client

Following the setup from the previous sections, you can configure Proxmox VE to use such pools to store VM and Container images. Simply use the GUI to

add a new [RBD](#) storage (see section [Ceph RADOS Block Devices \(RBD\)](#)).

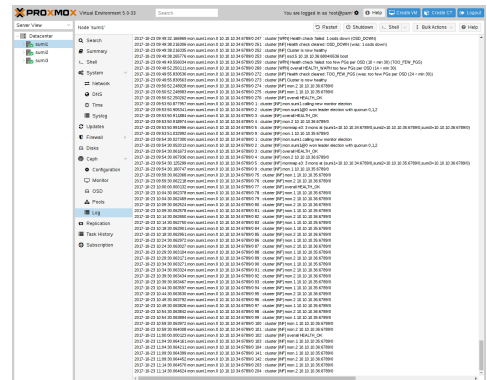
You also need to copy the keyring to a predefined location for an external Ceph cluster. If Ceph is installed on the Proxmox nodes itself, then this will be done automatically.

- The filename needs to be `<storage_id> + `.keyring`, where `<storage_id>` is the expression after `rbd:` in `/etc/pve/storage.cfg`. In the following example, `my-ceph-storage` is the `<storage_id>`:

```
mkdir /etc/pve/priv/ceph
cp /etc/ceph/ceph.client.admin.keyring
/etc/pve/priv/ceph/my-ceph-storage.keyring
```

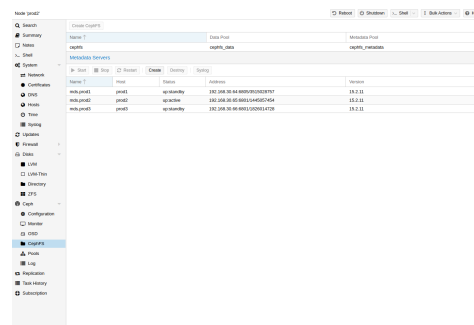
## CephFS

Ceph also provides a filesystem, which runs on top of the same object storage as RADOS block devices do. A **M**etadata **S**erver (**MDS**) is used to map the RADOS backed objects to files and directories, allowing Ceph to provide a POSIX-compliant,



replicated filesystem. This allows you to easily configure a clustered, highly available, shared filesystem. Ceph's Metadata Servers guarantee that files are evenly distributed over the entire Ceph cluster. As a result, even cases of high load will not overwhelm a single host, which can be an issue with traditional shared filesystem approaches, for example [NFS](#).

Proxmox VE supports both creating a hyper-converged CephFS and using an existing [CephFS as storage](#) to save backups, ISO files, and container templates.



## Metadata Server (MDS)

CephFS needs at least one Metadata Server to be configured and running, in order to function. You can create an MDS through the Proxmox VE web GUI's [Node](#) -> [CephFS](#) panel or from the command line with:

```
pveceph mds create
```

Multiple metadata servers can be created in a cluster, but with the default settings, only one can be active at a time. If an MDS or its node becomes unresponsive (or crashes), another [standby](#) MDS will get promoted to [active](#). You can speed up the handover between the active and standby MDS by using the [hotstandby](#) parameter option on creation, or if you have already created it you may set/add:

```
mds standby replay = true
```

in the respective MDS section of [/etc/pve/ceph.conf](#). With this enabled, the specified MDS will remain in a [warm](#) state, polling the active one, so that it can take over faster in case of any issues.



This active polling will have an additional performance impact on your system and the active [MDS](#).

## Multiple Active MDS

Since Luminous (12.2.x) you can have multiple active metadata servers running at once, but this is normally only useful if you have a high amount of clients running in parallel. Otherwise the MDS is rarely the bottleneck in a system. If you want to set this up, please refer to the Ceph documentation. <sup>[16]</sup>

## Create CephFS

With Proxmox VE's integration of CephFS, you can easily create a CephFS using the web interface, CLI or an external API interface. Some prerequisites are required for this to work:

### Prerequisites for a successful CephFS setup:

- [Install Ceph packages](#) - if this was already done some time ago, you may want to rerun it on an up-to-date system to ensure that all CephFS related packages get installed.
- [Setup Monitors](#)
- [Setup your OSDs](#)
- [Setup at least one MDS](#)

After this is complete, you can simply create a CephFS through either the Web GUI's [Node -> CephFS](#) panel or the command-line tool `pveceph`, for example:

```
pveceph fs create --pg_num 128 --  
add-storage
```

This creates a CephFS named *cephfs*, using a pool for its data named *cephfs\_data* with *128* placement groups and a pool for its metadata named *cephfs\_metadata* with one quarter of the data pool's placement groups (32). Check the [Proxmox VE managed Ceph pool chapter](#) or visit the Ceph documentation for more information regarding an appropriate placement group number (`pg_num`) for your setup <sup>[9]</sup>. Additionally, the `--add-storage` parameter will add the CephFS to the Proxmox VE storage configuration after it has been created successfully.

## Destroy CephFS



Destroying a CephFS will render all of its data unusable. This cannot be undone!

To completely and gracefully remove a CephFS, the following steps are necessary:

- Disconnect every non-Proxmox VE client (e.g. unmount the CephFS in guests).
- Disable all related CephFS Proxmox VE storage entries (to prevent it from being automatically mounted).
- Remove all used resources from guests (e.g. ISOs) that are on the CephFS you want to destroy.
- Unmount the CephFS storages on all cluster nodes manually with

```
umount /mnt/pve/<STORAGE-NAME>
```

Where `<STORAGE-NAME>` is the name of the CephFS storage in your Proxmox VE.

- Now make sure that no metadata server (MDS) is running for that CephFS, either by stopping or destroying them. This can be done through the web interface or via the command-line interface, for the latter you would issue the following command:

```
pveceph stop --service mds.NAME
```

to stop them, or

```
pveceph mds destroy NAME
```

to destroy them.

Note that standby servers will automatically be promoted to active when an active MDS is stopped or removed, so it is best to first stop all standby servers.

- Now you can destroy the CephFS with

```
pveceph fs destroy NAME --  
remove-storages --remove-pools
```

This will automatically destroy the underlying Ceph pools as well as remove the storages from pve config.

After these steps, the CephFS should be completely removed and if you have other CephFS instances, the stopped metadata servers can be started again to act as standbys.

## Ceph maintenance

---

## Replace OSDs

One of the most common maintenance tasks in Ceph is to replace the disk of an OSD. If a disk is already in a failed state, then you can go ahead and run through the steps in [Destroy OSDs](#). Ceph will recreate those copies on the remaining OSDs if possible. This rebalancing will start as soon as an OSD failure is detected or an OSD was actively stopped.

With the default `size/min_size (3/2)` of a pool, recovery only starts when `'size + 1'` nodes are available. The reason for this is that the Ceph object balancer [CRUSH](#) defaults to a full node as `'failure domain'`.

To replace a functioning disk from the GUI, go through the steps in [Destroy OSDs](#). The only addition is to wait until the cluster shows [HEALTH\\_OK](#) before stopping the OSD to destroy it.

On the command line, use the following commands:

```
ceph osd out osd.<id>
```

You can check with the command below if the OSD can be safely removed.

```
ceph osd safe-to-destroy osd.<id>
```

Once the above check tells you that it is safe to remove the OSD, you can continue with the following commands:

```
systemctl stop ceph-osd@<id>.service  
pveceph osd destroy <id>
```

Replace the old disk with the new one and use the same procedure as described in [Create OSDs](#).

## Trim/Discard

It is good practice to run [\*fstrim\*](#) (discard) regularly on VMs and containers. This releases data blocks that the filesystem isn't using anymore. It reduces data usage and resource load. Most modern operating systems issue such discard commands to their disks regularly. You only need to ensure that the Virtual Machines enable the [disk discard option](#).

## Scrub & Deep Scrub

Ceph ensures data integrity by *scrubbing* placement groups. Ceph checks every object in a PG for its health. There are two forms of Scrubbing, daily cheap metadata checks and weekly deep data checks. The weekly deep scrub reads the objects and uses checksums to ensure data integrity. If a running scrub interferes with business (performance) needs, you can adjust the time when scrubs <sup>[17]</sup> are executed.

## Ceph Monitoring and Troubleshooting

It is important to continuously monitor the health of a Ceph deployment from the beginning, either by using the Ceph tools or by accessing the status through the Proxmox VE [API](#).

The following Ceph commands can be used to see if the cluster is healthy (*HEALTH\_OK*), if there are warnings (*HEALTH\_WARN*), or even errors (*HEALTH\_ERR*). If the cluster is in an unhealthy state, the status commands below will also give you an overview of the current events and actions to take.

```
# single time output
pve# ceph -s
# continuously output status changes
# (press CTRL+C to stop)
pve# ceph -w
```

To get a more detailed view, every Ceph service has a log file under `/var/log/ceph/`. If more detail is required, the log level can be adjusted <sup>[18]</sup>.

You can find more information about troubleshooting <sup>[19]</sup> a Ceph cluster on the official website.

- 
1. Ceph intro <https://docs.ceph.com/en/quincy/start/intro/>
  2. Ceph architecture <https://docs.ceph.com/en/quincy/architecture/>
  3. Ceph glossary <https://docs.ceph.com/en/quincy/glossary>
  4. Full Mesh Network for Ceph [https://pve.proxmox.com/wiki/Full\\_Mesh\\_Network\\_for\\_Ceph\\_Server](https://pve.proxmox.com/wiki/Full_Mesh_Network_for_Ceph_Server)
  5. Ceph Monitor <https://docs.ceph.com/en/quincy/start/intro/>
  6. Ceph Manager <https://docs.ceph.com/en/quincy/mgr/>
  7. Ceph Bluestore <https://ceph.com/community/new-luminous-bluestore/>
  8. PG calculator <https://web.archive.org/web/20210301111112/http://ceph.com/pgcalc/>



**9. Placement Groups**

<https://docs.ceph.com/en/quincy/rados/operations/placement-groups/>

**10. Automated Scaling**

<https://docs.ceph.com/en/quincy/rados/operations/placement-groups/#automated-scaling>

**11. Ceph pool operation**

<https://docs.ceph.com/en/quincy/rados/operations/pools/>

**12. Ceph Erasure Coded Pool Recovery**

<https://docs.ceph.com/en/quincy/rados/operations/erasure-code/#erasure-coded-pool-recovery>

**13. Ceph Erasure Code Profile**

<https://docs.ceph.com/en/quincy/rados/operations/erasure-code/#erasure-code-profiles>

**14. CRUSH** <https://ceph.com/wp-content/uploads/2016/08/weil-crush-sco6.pdf>**15. CRUSH map**

<https://docs.ceph.com/en/quincy/rados/operations/crush-map/>

**16. Configuring multiple active MDS daemons**

<https://docs.ceph.com/en/quincy/cephfs/multimds/>

**17. Ceph scrubbing**

<https://docs.ceph.com/en/quincy/rados/configuration/osd-config-ref/#scrubbing>

**18. Ceph log and debugging**

<https://docs.ceph.com/en/quincy/rados/troubleshooting/log-and-debug/>

**19. Ceph troubleshooting**

<https://docs.ceph.com/en/quincy/rados/troubleshooting/>

---

Version 8.2.2

Last updated Thu Apr 25 09:24:16 CEST 2024