Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

WELCOME TO CEPH ¶

Ceph uniquely delivers **object, block, and file storage in one unified system**.

CEPH OBJECT STORE

- RESTful Interface
- S3- and Swift-compliant APIs
- S3-style subdomains
- Unified S3/Swift namespace
- User management
- Usage tracking
- Striped objects
- Cloud solution integration
- Multi-site deployment
- Multi-site replication

See Ceph Object Store for additional details.

CEPH BLOCK DEVICE

- Thin-provisioned
- Images up to 16 exabytes
- Configurable striping
- In-memory caching
- Snapshots
- Copy-on-write cloning
- Kernel driver support
- KVM/libvirt support
- Back-end for cloud solutions
- Incremental backup
- Disaster recovery (multisite asynchronous replication)

See Ceph Block Device for additional details.

CEPH FILE SYSTEM

- POSIX-compliant semantics
- Separates metadata from data
- Dynamic rebalancing
- Subdirectory snapshots
- Configurable striping
- Kernel driver support
- FUSE support
- NFS/CIFS deployable
- Use with Hadoop (replace HDFS)

See Ceph File System for additional details.

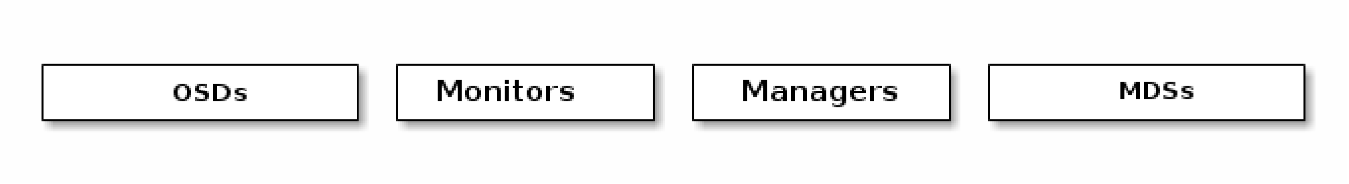
Ceph is highly reliable, easy to manage, and free. The power of Ceph can transform your company's IT infrastructure and your ability to manage vast amounts of data. To try Ceph, see our Getting Started guides. To learn more about Ceph, see our Architecture section.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

INTRO TO CEPH

Whether you want to provide Ceph Object Storage and/or Ceph Block Device services to Cloud Platforms, deploy a Ceph File System or use Ceph for another purpose, all Ceph Storage Cluster deployments begin with setting up each Ceph Node, your network, and the Ceph Storage Cluster. A Ceph Storage Cluster requires at least one Ceph Monitor, Ceph Manager, and Ceph OSD (Object Storage Daemon). The Ceph Metadata Server is also required when running Ceph File System clients.



- **Monitors**: A Ceph Monitor (`ceph-mon`) maintains maps of the cluster state, including the monitor map, manager map, the OSD map, and the CRUSH map. These maps are critical cluster state required for Ceph daemons to coordinate with each other. Monitors are also responsible for managing authentication between daemons and clients. At least three monitors are normally required for redundancy and high availability.
- **Managers**: A Ceph Manager daemon (`ceph-mgr`) is responsible for keeping track of runtime metrics and the current state of the Ceph cluster, including storage utilization, current performance metrics, and system load. The Ceph Manager daemons also host python-based modules to manage and expose Ceph cluster information, including a web-based Ceph Dashboard and REST API. At least two managers are normally required for high availability.
- **Ceph OSDs**: A Ceph OSD (object storage daemon, `ceph-osd`) stores data, handles data replication, recovery, rebalancing, and provides some monitoring information to Ceph Monitors and Managers by checking other Ceph OSD Daemons for a heartbeat. At least 3 Ceph OSDs are normally required for redundancy and high availability.
- **MDSs**: A Ceph Metadata Server (MDS, `ceph-mds`) stores metadata on behalf of the Ceph File System (i.e., Ceph Block Devices and Ceph Object Storage do not use MDS). Ceph Metadata Servers allow POSIX file system users to execute basic commands (like `ls`, `find`, etc.) without placing an enormous burden on the Ceph Storage Cluster.

Ceph stores data as objects within logical storage pools. Using the CRUSH algorithm, Ceph calculates which placement group should contain the object, and further calculates which Ceph OSD Daemon should store the placement group. The CRUSH algorithm enables the Ceph Storage Cluster to scale, rebalance, and recover dynamically.

RECOMMENDATIONS

GET INVOLVED

To begin using Ceph in production, you should review our hardware recommendations and operating system recommendations.

- Hardware Recommendations
  - CPU
  - RAM
  - Data Storage
  - Networks
  - Failure Domains
  - Minimum Hardware Recommendations
  - Production Cluster Examples
- OS Recommendations
  - Ceph Dependencies
  - Platforms

You can avail yourself of help or contribute documentation, source code or bugs by getting involved in the Ceph community.

- Get Involved in the Ceph Community!
- Documenting Ceph
  - Making Contributions
  - Documentation Style Guide

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

HARDWARE RECOMMENDATIONS

Ceph was designed to run on commodity hardware, which makes building and maintaining petabyte-scale data clusters economically feasible. When planning out your cluster hardware, you will need to balance a number of considerations, including failure domains and potential performance issues. Hardware planning should include distributing Ceph daemons and other processes that use Ceph across many hosts. Generally, we recommend running Ceph daemons of a specific type on a host configured for that type of daemon. We recommend using other hosts for processes that utilize your data cluster (e.g., OpenStack, CloudStack, etc).

Tip: Check out the Ceph blog too.

CPU

Ceph metadata servers dynamically redistribute their load, which is CPU intensive. So your metadata servers should have significant processing power (e.g., quad core or better CPUs). Ceph OSDs run the RADOS service, calculate data placement with CRUSH, replicate data, and maintain their own copy of the cluster map. Therefore, OSDs should have a reasonable amount of processing power (e.g., dual core processors). Monitors simply maintain a master copy of the cluster map, so they are not CPU intensive. You must also consider whether the host machine will run CPU-intensive processes in addition to Ceph daemons. For example, if your hosts will run computing VMs (e.g., OpenStack Nova), you will need to ensure that these other processes leave sufficient processing power for Ceph daemons. We recommend running additional CPU-intensive processes on separate hosts.

RAM

Generally, more RAM is better.

MONITORS AND MANAGERS (CEPH-MON AND CEPH-MGR)

Monitor and manager daemon memory usage generally scales with the size of the cluster. For small clusters, 1-2 GB is generally sufficient. For large clusters, you should provide more (5-10 GB). You may also want to consider tuning settings like `mon_osd_cache_size` or `rocksdb_cache_size`.

METADATA SERVERS (CEPH-MDS)

The metadata daemon memory utilization depends on how much memory its cache is configured to consume. We recommend 1 GB as a minimum for most systems. See `mds_cache_memory`.

OSDS (CEPH-OSD)

By default, OSDs that use the BlueStore backend require 3-5 GB of RAM. You can adjust the amount of memory the OSD consumes with the `osd_memory_target` configuration option when BlueStore is in use. When using the legacy FileStore backend, the operating system page cache is used for caching data, so no tuning is normally needed, and the OSD memory consumption is generally related to the number of PGs per daemon in the system.

DATA STORAGE

Plan your data storage configuration carefully. There are significant cost and performance tradeoffs to consider when planning for data storage. Simultaneous OS operations, and simultaneous request for read and write operations from multiple daemons against a single drive can slow performance considerably.

Important: Since Ceph has to write all data to the journal before it can send an ACK (for XFS at least), having the journal and OSD performance in balance is really important!

HARD DISK DRIVES

OSDs should have plenty of hard disk drive space for object data. We recommend a minimum hard disk drive size of 1 terabyte. Consider the cost-per-gigabyte advantage of larger disks. We recommend dividing the price of the hard disk drive by the number of gigabytes to arrive at a cost per gigabyte, because larger drives may have a significant impact on the cost-per-gigabyte. For example, a 1 terabyte hard disk priced at $75.00 has a cost of $0.07 per gigabyte (i.e., $75 / 1024 = 0.0732). By contrast, a 3 terabyte hard disk priced at $150.00 has a cost of $0.05 per gigabyte (i.e., $150 / 3072 = 0.0488). In the foregoing example, using the 1 terabyte disks would generally increase the cost per gigabyte by 40%–rendering your cluster substantially less cost efficient. Also, the larger the storage drive capacity, the more memory per Ceph OSD Daemon you will need, especially during rebalancing, backfilling and recovery. A general rule of thumb is ~1GB of RAM for 1TB of storage space.

Tip: Running multiple OSDs on a single disk–irrespective of partitions–is **NOT** a good idea.

Tip: Running an OSD and a monitor or a metadata server on a single disk–irrespective of partitions–is **NOT** a good idea either.

Storage drives are subject to limitations on seek time, access time, read and write times, as well as total throughput. These physical limitations affect overall system performance–especially during recovery. We recommend using a dedicated drive for the operating system and software, and one drive for each Ceph OSD Daemon you run on the host. Most “slow OSD” issues arise due to running an operating system, multiple OSDs, and/or multiple journals on the same drive. Since the cost of troubleshooting performance issues on a small cluster likely exceeds the cost of the extra disk drives, you can optimize your cluster design planning by avoiding the temptation to overtax the OSD storage drives.

You may run multiple Ceph OSD Daemons per hard disk drive, but this will likely lead to resource contention and diminish the overall throughput. You may store a journal and object data on the same drive, but this may increase the time it takes to journal a write and ACK to the client. Ceph must write to the journal before it can ACK the write.

Ceph best practices dictate that you should run operating systems, OSD data and OSD journals on separate drives.

SOLID STATE DRIVES

One opportunity for performance improvement is to use solid-state drives (SSDs) to reduce random access time and read latency while accelerating throughput. SSDs often cost more than 10x as much per gigabyte when compared to a hard disk drive, but SSDs often exhibit access times that are at least 100x faster than a hard disk drive.

SSDs do not have moving mechanical parts so they are not necessarily subject to the same types of limitations as hard disk drives. SSDs do have significant limitations though. When evaluating SSDs, it is important to consider the performance of sequential reads and writes. An SSD that has 400MB/s sequential write throughput may have much better performance than an SSD with 120MB/s of sequential write throughput when storing multiple journals for multiple OSDs.

> **Important:** We recommend exploring the use of SSDs to improve performance. However, before making a significant investment in SSDs, we **strongly recommend** both reviewing the performance metrics of an SSD and testing the SSD in a test configuration to gauge performance.

Since SSDs have no moving mechanical parts, it makes sense to use them in the areas of Ceph that do not use a lot of storage space (e.g., journals). Relatively inexpensive SSDs may appeal to your sense of economy. Use caution. Acceptable IOPS are not enough when selecting an SSD for use with Ceph. There are a few important performance considerations for journals and SSDs:

- **Write-intensive semantics:** Journaling involves write-intensive semantics, so you should ensure that the SSD you choose to deploy will perform equal to or better than a hard disk drive when writing data. Inexpensive SSDs may introduce write latency even as they accelerate access time, because sometimes high performance hard drives can write as fast or faster than some of the more economical SSDs available on the market!
- **Sequential Writes:** When you store multiple journals on an SSD you must consider the sequential write limitations of the SSD too, since they may be handling requests to write to multiple OSD journals simultaneously.
- **Partition Alignment:** A common problem with SSD performance is that people like to partition drives as a best practice, but they often overlook proper partition alignment with SSDs, which can cause SSDs to transfer data much more slowly. Ensure that SSD partitions are properly aligned.

While SSDs are cost prohibitive for object storage, OSDs may see a significant performance improvement by storing an OSD’s journal on an SSD and the OSD’s object data on a separate hard disk drive. The `osd` `journal` configuration setting defaults to `/var/lib/ceph/osd/$cluster-$id/journal`. You can mount this path to an SSD or to an SSD partition so that it is not merely a file on the same disk as the object data.

One way Ceph accelerates CephFS file system performance is to segregate the storage of CephFS metadata from the storage of the CephFS file contents. Ceph provides a default `metadata` pool for CephFS metadata. You will never have to create a pool for CephFS metadata, but you can create a CRUSH map hierarchy for your CephFS metadata pool that points only to a host's SSD storage media. See Mapping Pools to Different Types of OSDs for details.

CONTROLLERS

Disk controllers also have a significant impact on write throughput. Carefully, consider your selection of disk controllers to ensure that they do not create a performance bottleneck.

Tip: The Ceph blog is often an excellent source of information on Ceph performance issues. See Ceph Write Throughput 1 and Ceph Write Throughput 2 for additional details.

ADDITIONAL CONSIDERATIONS

You may run multiple OSDs per host, but you should ensure that the sum of the total throughput of your OSD hard disks doesn't exceed the network bandwidth required to service a client's need to read or write data. You should also consider what percentage of the overall data the cluster stores on each host. If the percentage on a particular host is large and the host fails, it can lead to problems such as exceeding the `full ratio`, which causes Ceph to halt operations as a safety precaution that prevents data loss.

When you run multiple OSDs per host, you also need to ensure that the kernel is up to date. See OS Recommendations for notes on `glibc` and `syncfs(2)` to ensure that your hardware performs as expected when running multiple OSDs per host.

NETWORKS

We recommend that each host has at least two 1Gbps network interface controllers (NICs). Since most commodity hard disk drives have a throughput of approximately 100MB/second, your NICs should be able to handle the traffic for the OSD disks on your host. We recommend a minimum of two NICs to account for a public (front-side) network and a cluster (back-side) network. A cluster network (preferably not connected to the internet) handles the additional load for data replication and helps stop denial of service attacks that prevent the cluster from achieving `active + clean` states for placement groups as OSDs replicate data across the cluster. Consider starting with a 10Gbps network in your racks. Replicating 1TB of data across a 1Gbps network takes 3 hours, and 3TBs (a typical drive configuration) takes 9 hours. By contrast, with a 10Gbps network, the replication times would be 20 minutes and 1 hour respectively. In a petabyte-scale cluster, failure of an OSD disk should be an expectation, not an exception. System administrators will appreciate PGs recovering from a `degraded` state to an `active + clean` state as rapidly as possible, with price / performance tradeoffs taken into consideration. Additionally, some deployment tools (e.g., Dell's

Crowbar) deploy with five different networks, but employ VLANs to make hardware and network cabling more manageable. VLANs using 802.1q protocol require VLAN-capable NICs and Switches. The added hardware expense may be offset by the operational cost savings for network setup and maintenance. When using VLANs to handle VM traffic between the cluster and compute stacks (e.g., OpenStack, CloudStack, etc.), it is also worth considering using 10G Ethernet. Top-of-rack routers for each network also need to be able to communicate with spine routers that have even faster throughput–e.g., 40Gbps to 100Gbps.

Your server hardware should have a Baseboard Management Controller (BMC). Administration and deployment tools may also use BMCs extensively, so consider the cost/benefit tradeoff of an out-of-band network for administration. Hypervisor SSH access, VM image uploads, OS image installs, management sockets, etc. can impose significant loads on a network. Running three networks may seem like overkill, but each traffic path represents a potential capacity, throughput and/or performance bottleneck that you should carefully consider before deploying a large scale data cluster.

FAILURE DOMAINS

A failure domain is any failure that prevents access to one or more OSDs. That could be a stopped daemon on a host; a hard disk failure, an OS crash, a malfunctioning NIC, a failed power supply, a network outage, a power outage, and so forth. When planning out your hardware needs, you must balance the temptation to reduce costs by placing too many responsibilities into too few failure domains, and the added costs of isolating every potential failure domain.

MINIMUM HARDWARE RECOMMENDATIONS

Ceph can run on inexpensive commodity hardware. Small production clusters and development clusters can run successfully with modest hardware.

Process	Criteria	Minimum Recommended
`ceph-osd`	Processor	• 1x 64-bit AMD-64 • 1x 32-bit ARM dual-core or better
	RAM	~1GB for 1TB of storage per daemon
	Volume Storage	1x storage drive per daemon
	Journal	1x SSD partition per daemon (optional)
	Network	2x 1GB Ethernet NICs
`ceph-mon`	Processor	• 1x 64-bit AMD-64 • 1x 32-bit ARM dual-core or better
	RAM	1 GB per daemon
	Disk Space	10 GB per daemon
	Network	2x 1GB Ethernet NICs

Process	Criteria	Minimum Recommended
`ceph-mds`	Processor	1x 64-bit AMD-64 quad-core 1x 32-bit ARM quad-core
	RAM	1 GB minimum per daemon
	Disk Space	1 MB per daemon
	Network	2x 1GB Ethernet NICs

Tip: If you are running an OSD with a single disk, create a partition for your volume storage that is separate from the partition containing the OS. Generally, we recommend separate disks for the OS and the volume storage.

PRODUCTION CLUSTER EXAMPLES

Production clusters for petabyte scale data storage may also use commodity hardware, but should have considerably more memory, processing power and data storage to account for heavy traffic loads.

DELL EXAMPLE

A recent (2012) Ceph cluster project is using two fairly robust hardware configurations for Ceph OSDs, and a lighter configuration for monitors.

Configuration	Criteria	Minimum Recommended
Dell PE R510	Processor	2x 64-bit quad-core Xeon CPUs
	RAM	16 GB
	Volume Storage	8x 2TB drives. 1 OS, 7 Storage
	Client Network	2x 1GB Ethernet NICs
	OSD Network	2x 1GB Ethernet NICs
	Mgmt. Network	2x 1GB Ethernet NICs
Dell PE R515	Processor	1x hex-core Opteron CPU
	RAM	16 GB
	Volume Storage	12x 3TB drives. Storage
	OS Storage	1x 500GB drive. Operating System.
	Client Network	2x 1GB Ethernet NICs
	OSD Network	2x 1GB Ethernet NICs
	Mgmt. Network	2x 1GB Ethernet NICs

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

OS RECOMMENDATIONS

CEPH DEPENDENCIES

As a general rule, we recommend deploying Ceph on newer releases of Linux. We also recommend deploying on releases with long-term support.

LINUX KERNEL

- **Ceph Kernel Client**

  If you are using the kernel client to map RBD block devices or mount CephFS, the general advice is to use a “stable” or “longterm maintenance” kernel series provided by either http://kernel.org or your Linux distribution on any client hosts.

  For RBD, if you choose to *track* long-term kernels, we currently recommend 4.x-based “longterm maintenance” kernel series:

  - 4.19.z
  - 4.14.z

  For CephFS, see CephFS best practices for kernel version guidance.

  Older kernel client versions may not support your CRUSH tunables profile or other newer features of the Ceph cluster, requiring the storage cluster to be configured with those features disabled.

PLATFORMS

The charts below show how Ceph’s requirements map onto various Linux platforms. Generally speaking, there is very little dependence on specific distributions aside from the kernel and system initialization package (i.e., sysvinit, upstart, systemd).

NAUTILUS (14.2.Z)

Distro	Release	Code Name	Kernel	Notes	Testing
CentOS	7	N/A	linux-3.10.0	3	B, I, C

Distro	Release	Code Name	Kernel	Notes	Testing
Debian	8.0	Jessie	linux-3.16.0	1, 2	B, I
Debian	9.0	Stretch	linux-4.9	1, 2	B, I
RHEL	7	Maipo	linux-3.10.0		B, I
Ubuntu	14.04	Trusty Tahr	linux-3.13.0		B, I, C
Ubuntu	16.04	Xenial Xerus	linux-4.4.0	3	B, I, C
Ubuntu	18.04	Bionic Beaver	linux-4.15	3	B, I, C
openSUSE	15.1	Leap	linux-4.12		
openSUSE		Tumbleweed	linux-5.1.7		

LUMINOUS (12.2.Z)

Distro	Release	Code Name	Kernel	Notes	Testing
CentOS	7	N/A	linux-3.10.0	3	B, I, C
Debian	8.0	Jessie	linux-3.16.0	1, 2	B, I
Debian	9.0	Stretch	linux-4.9	1, 2	B, I
Fedora	22	N/A	linux-3.14.0		B, I
RHEL	7	Maipo	linux-3.10.0		B, I
Ubuntu	14.04	Trusty Tahr	linux-3.13.0		B, I, C
Ubuntu	16.04	Xenial Xerus	linux-4.4.0	3	B, I, C

NOTES

- **1**: The default kernel has an older version of `btrfs` that we do not recommend for `ceph-osd` storage nodes. We recommend using `XFS`.
- **2**: The default kernel has an old Ceph client that we do not recommend for kernel client (kernel RBD or the Ceph file system). Upgrade to a recommended kernel.
- **3**: The default kernel regularly fails in QA when the `btrfs` file system is used. We do not recommend using `btrfs` for backing Ceph OSDs.

TESTING

- **B**: We build release packages for this platform. For some of these platforms, we may also continuously build all ceph branches and exercise basic unit tests.
- **I**: We do basic installation and functionality tests of releases on this platform.
- **C**: We run a comprehensive functional, regression, and stress test suite on this platform on a continuous basis. This includes development branches, pre-release, and released code.

Notice: This document is for a development version of Ceph.

Edit on GitHub | Report a Documentation Bug

GET INVOLVED IN THE CEPH COMMUNITY!

These are exciting times in the Ceph community! Get involved!

Channel	Description	Contact Info
Blog	Check the Ceph Blog periodically to keep track of Ceph progress and important announcements.	http://ceph.com/community/blog/
Planet Ceph	Check the blog aggregation on Planet Ceph for interesting stories, information and experiences from the community.	https://ceph.com/category/planet/
Wiki	Check the Ceph Wiki is a source for more community and development related topics. You can find there information about blueprints, meetups, the Ceph Developer Summits and more.	http://wiki.ceph.com/
IRC	As you delve into Ceph, you may have questions or feedback for the Ceph development team. Ceph developers are often available on the `#ceph` IRC channel particularly during daytime hours in the US Pacific Standard Time zone. While `#ceph` is a good starting point for cluster operators and users, there is also `#ceph-devel` and `#ceph-dashboard` dedicated for Ceph developers.	• **Domain:** `irc.oftc.net` • **Channels:** `#ceph`, `#ceph-devel`, `#ceph-dashboard`
User List	Ask and answer user-related questions by subscribing to the email list at ceph-users@ceph.io. You can opt out of the email list at any time by unsubscribing. A simple email is all it takes!	• User Subscribe • User Unsubscribe • User Archives

Channel	Description	Contact Info
Devel List	Keep in touch with developer activity by subscribing to the email list at dev@ceph.io. You can opt out of the email list at any time by unsubscribing. A simple email is all it takes!	- Devel Subscribe - Devel Unsubscribe - Devel Archives
Commit List	Subscribe to ceph-commit@ceph.com to get commit notifications via email. You can opt out of the email list at any time by unsubscribing. A simple email is all it takes!	- Commit Subscribe - Commit Unsubscribe - Mailing list archives
QA List	For Quality Assurance (QA) related activities subscribe to this list. You can opt out of the email list at any time by unsubscribing. A simple email is all it takes!	- QA Subscribe - QA Unsubscribe - Mailing list archives
Community List	For all discussions related to the Ceph User Committee and other community topics. You can opt out of the email list at any time by unsubscribing. A simple email is all it takes!	- Community Subscribe - Community Unsubscribe - Mailing list archives
Bug Tracker	You can help keep Ceph production worthy by filing and tracking bugs, and providing feature requests using the Bug Tracker.	http://tracker.ceph.com/projects/ceph
Source Code	If you would like to participate in development, bug fixing, or if you just want the very latest code for Ceph, you can get it at http://github.com. See Ceph Source Code for details on cloning from github.	- http://github.com/ceph/ceph - http://download.ceph.com/tarballs/
Ceph Calendar	Learn about upcoming Ceph events.	https://ceph.io/contribute/

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

DOCUMENTING CEPH

The **easiest way** to help the Ceph project is to contribute to the documentation. As the Ceph user base grows and the development pace quickens, an increasing number of people are updating the documentation and adding new information. Even small contributions like fixing spelling errors or clarifying instructions will help the Ceph project immensely.

The Ceph documentation source resides in the `ceph/doc` directory of the Ceph repository, and Python Sphinx renders the source into HTML and manpages. The http://ceph.com/docs link currently displays the `master` branch by default, but you may view documentation for older branches (e.g., `argonaut`) or future branches (e.g., `next`) as well as work-in-progress branches by substituting `master` with the branch name you prefer.

MAKING CONTRIBUTIONS

Making a documentation contribution generally involves the same procedural sequence as making a code contribution, except that you must build documentation source instead of compiling program source. The sequence includes the following steps:

1. Get the Source
2. Select a Branch
3. Make a Change
4. Build the Source
5. Commit the Change
6. Push the Change
7. Make a Pull Request
8. Notify Us

GET THE SOURCE

Ceph documentation lives in the Ceph repository right alongside the Ceph source code under the `ceph/doc` directory. For details on github and Ceph, see Get Involved in the Ceph Community!.

The most common way to make contributions is to use the Fork and Pull approach. You must:

1. Install git locally. For Debian/Ubuntu, execute:

```
sudo apt-get install git
```

For Fedora, execute:

```
sudo yum install git
```

For CentOS/RHEL, execute:

```
sudo yum install git
```

2. Ensure your `.gitconfig` file has your name and email address.

```
[user]
   email = {your-email-address}
   name = {your-name}
```

   For example:

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
```

3. Create a github account (if you don’t have one).
4. Fork the Ceph project. See https://github.com/ceph/ceph.
5. Clone your fork of the Ceph project to your local host.

Ceph organizes documentation into an information architecture primarily by its main components.

- **Ceph Storage Cluster:** The Ceph Storage Cluster documentation resides under the `doc/rados` directory.
- **Ceph Block Device:** The Ceph Block Device documentation resides under the `doc/rbd` directory.
- **Ceph Object Storage:** The Ceph Object Storage documentation resides under the `doc/radosgw` directory.
- **Ceph File System:** The Ceph File System documentation resides under the `doc/cephfs` directory.
- **Installation (Quick):** Quick start documentation resides under the `doc/start` directory.
- **Installation (Manual):** Manual installation documentation resides under the `doc/install` directory.
- **Manpage:** Manpage source resides under the `doc/man` directory.
- **Developer:** Developer documentation resides under the `doc/dev` directory.
- **Images:** If you include images such as JPEG or PNG files, you should store them under the `doc/images` directory.

SELECT A BRANCH

When you make small changes to the documentation, such as fixing typographical errors or clarifying explanations, use the `master` branch (default). You should also use the `master` branch when making contributions to features that are in the current release. `master` is the most commonly used branch.

```
git checkout master
```

When you make changes to documentation that affect an upcoming release, use the `next` branch. `next` is the second most commonly used branch.

```
git checkout next
```

When you are making substantial contributions such as new features that are not yet in the current release; if your contribution is related to an issue with a tracker ID; or, if you want to see your documentation rendered on the Ceph.com website before it gets merged into the `master` branch, you should create a branch. To distinguish branches that include only documentation updates, we prepend them with `wip-doc` by convention, following the form `wip-doc-{your-branch-name}`. If the branch relates to an issue filed in http://tracker.ceph.com/issues, the branch name incorporates the issue number. For example, if a documentation branch is a fix for issue #4000, the branch name should be `wip-doc-4000` by convention and the relevant tracker URL will be http://tracker.ceph.com/issues/4000.

> **Note:** Please do not mingle documentation contributions and source code contributions in a single commit. When you keep documentation commits separate from source code commits, it simplifies the review process. We highly recommend that any pull request that adds a feature or a configuration option, should also include a documentation commit, describing the relevant changes/options.

Before you create your branch name, ensure that it doesn't already exist in the local or remote repository.

```
git branch -a | grep wip-doc-{your-branch-name}
```

If it doesn't exist, create your branch:

```
git checkout -b wip-doc-{your-branch-name}
```

MAKE A CHANGE

Modifying a document involves opening a restructuredText file, changing its contents, and saving the changes. See Documentation Style Guide for details on syntax requirements.

Adding a document involves creating a new restructuredText file under the `doc` directory or its subdirectories and saving the file with a `*.rst` file extension. You must also include a reference to the document: a hyperlink or a table of contents entry. The `index.rst` file of a top-level directory usually contains a TOC, where you can add the new file name. All documents must have a title. See Headings for details.

Your new document doesn't get tracked by `git` automatically. When you want to add the document to the repository, you must use `git add {path-to-filename}`. For example, from the top level directory of the repository, adding an `example.rst` file to the `rados` subdirectory would look like this:

```
git add doc/rados/example.rst
```

Deleting a document involves removing it from the repository with `git rm {path-to-filename}`. For example:

```
git rm doc/rados/example.rst
```

You must also remove any reference to a deleted document from other documents.

BUILD THE SOURCE

To build the documentation, navigate to the `ceph` repository directory:

```
cd ceph
```

To build the documentation on Debian/Ubuntu, Fedora, or CentOS/RHEL, execute:

```
admin/build-doc
```

To scan for the reachability of external links, execute:

```
admin/build-doc linkcheck
```

Executing `admin/build-doc` will create a `build-doc` directory under `ceph`. You may need to create a directory under `ceph/build-doc` for output of Javadoc files.

```
mkdir -p output/html/api/libcephfs-java/javadoc
```

The build script `build-doc` will produce an output of errors and warnings. You MUST fix errors in documents you modified before committing a change, and you SHOULD fix warnings that are related to syntax you modified.

Important: You must validate ALL HYPERLINKS. If a hyperlink is broken, it automatically breaks the build!

Once you build the documentation set, you may start an HTTP server at `http://localhost:8080/` to view it:

```
admin/serve-doc
```

You can also navigate to `build-doc/output` to inspect the built documents. There should be an `html` directory and a `man` directory containing documentation in HTML and manpage formats respectively.

BUILD THE SOURCE (FIRST TIME)

Ceph uses Python Sphinx, which is generally distribution agnostic. The first time you build Ceph documentation, it will generate a doxygen XML tree, which is a bit time consuming.

Python Sphinx does have some dependencies that vary across distributions. The first time you build the documentation, the script will notify you if you do not have the dependencies installed. To run Sphinx and build documentation successfully, the following packages are required:

DEBIAN/UBUNTU

- gcc
- python-dev
- python-pip
- python-virtualenv
- python-sphinx
- libxml2-dev
- libxslt1-dev
- doxygen
- graphviz
- ant
- ditaa

FEDORA

- gcc
- python-devel
- python-pip
- python-virtualenv
- python-docutils
- python-jinja2
- python-pygments
- python-sphinx
- libxml2-devel
- libxslt1-devel
- doxygen
- graphviz
- ant
- ditaa

CENTOS/RHEL

- gcc
- python-devel
- python-pip
- python-virtualenv
- python-docutils
- python-jinja2
- python-pygments
- python-sphinx
- libxml2-dev
- libxslt1-dev
- doxygen
- graphviz
- ant

Install each dependency that is not installed on your host. For Debian/Ubuntu distributions, execute the following:

```
sudo apt-get install gcc python-dev python-pip python-virtualenv li
sudo apt-get install python-sphinx
```

For Fedora distributions, execute the following:

```
sudo yum install gcc python-devel python-pip python-virtualenv libx
sudo pip install html2text
sudo yum install python-jinja2 python-pygments python-docutils pyth
sudo yum install jericho-html ditaa
```

For CentOS/RHEL distributions, it is recommended to have `epel` (Extra Packages for Enterprise Linux) repository as it provides some extra packages which are not available in the default repository. To install `epel`, execute the following:

```
sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-rele
```

For CentOS/RHEL distributions, execute the following:

```
sudo yum install gcc python-devel python-pip python-virtualenv libx
sudo pip install html2text
```

For CentOS/RHEL distributions, the remaining python packages are not available in the default and `epel` repositories. So, use http://rpmfind.net/ to find the packages. Then, download them from a mirror and install them. For example:

```
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/python-ji
sudo yum install python-jinja2-2.7.2-2.el7.noarch.rpm
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/python-py
sudo yum install python-pygments-1.4-9.el7.noarch.rpm
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/python-do
sudo yum install python-docutils-0.11-0.2.20130715svn7687.el7.noarc
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/python-sp
sudo yum install python-sphinx-1.1.3-11.el7.noarch.rpm
```

Ceph documentation makes extensive use of ditaa, which is not presently built for CentOS/RHEL7. You must install `ditaa` if you are making changes to `ditaa` diagrams so that you can verify that they render properly before you commit new or modified `ditaa` diagrams. You may retrieve compatible required packages for CentOS/RHEL distributions and install them manually. To run `ditaa` on CentOS/RHEL7, following dependencies are required:

- jericho-html
- jai-imageio-core
- batik

Use http://rpmfind.net/ to find compatible `ditaa` and the dependencies. Then, download them from a mirror and install them. For example:

```
wget http://rpmfind.net/linux/fedora/linux/releases/22/Everything/x
sudo yum install jericho-html-3.3-4.fc22.noarch.rpm
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/jai-image
sudo yum install jai-imageio-core-1.2-0.14.20100217cvs.el7.noarch.r
wget http://rpmfind.net/linux/centos/7/os/x86_64/Packages/batik-1.8
sudo yum install batik-1.8-0.12.svn1230816.el7.noarch.rpm
```

```
wget http://rpmfind.net/linux/fedora/linux/releases/22/Everything/x
sudo yum install ditaa-0.9-13.r74.fc21.noarch.rpm
```

Once you have installed all these packages, build the documentation by following the steps given in Build the Source.

COMMIT THE CHANGE

Ceph documentation commits are simple, but follow a strict convention:

- A commit SHOULD have 1 file per commit (it simplifies rollback). You MAY commit multiple files with related changes. Unrelated changes SHOULD NOT be put into the same commit.
- A commit MUST have a comment.
- A commit comment MUST be prepended with `doc:`. (strict)
- The comment summary MUST be one line only. (strict)
- Additional comments MAY follow a blank line after the summary, but should be terse.
- A commit MAY include `Fixes: #{bug number}`.
- Commits MUST include `Signed-off-by: Firstname Lastname <email>`. (strict)

Tip: Follow the foregoing convention particularly where it says `(strict)` or you will be asked to modify your commit to comply with this convention.

The following is a common commit comment (preferred):

```
doc: Fixes a spelling error and a broken hyperlink.

Signed-off-by: John Doe <john.doe@gmail.com>
```

The following comment includes a reference to a bug.

```
doc: Fixes a spelling error and a broken hyperlink.

Fixes: #1234

Signed-off-by: John Doe <john.doe@gmail.com>
```

The following comment includes a terse sentence following the comment summary. There is a carriage return between the summary line and the description:

```
doc: Added mon setting to monitor config reference

Describes 'mon setting', which is a new setting added
to config_opts.h.
```

```
Signed-off-by: John Doe <john.doe@gmail.com>
```

To commit changes, execute the following:

```
git commit -a
```

An easy way to manage your documentation commits is to use visual tools for `git`. For example, `gitk` provides a graphical interface for viewing the repository history, and `git-gui` provides a graphical interface for viewing your uncommitted changes, staging them for commit, committing the changes and pushing them to your forked Ceph repository.

For Debian/Ubuntu, execute:

```
sudo apt-get install gitk git-gui
```

For Fedora/CentOS/RHEL, execute:

```
sudo yum install gitk git-gui
```

Then, execute:

```
cd {git-ceph-repo-path}
gitk
```

Finally, select **File->Start git gui** to activate the graphical user interface.

PUSH THE CHANGE

Once you have one or more commits, you must push them from the local copy of the repository to `github`. A graphical tool like `git-gui` provides a user interface for pushing to the repository. If you created a branch previously:

```
git push origin wip-doc-{your-branch-name}
```

Otherwise:

```
git push
```

MAKE A PULL REQUEST

As noted earlier, you can make documentation contributions using the Fork and Pull approach.

NOTIFY US

After you make a pull request, please email ceph-docs@redhat.com.

DOCUMENTATION STYLE GUIDE

One objective of the Ceph documentation project is to ensure the readability of the documentation in both native restructuredText format and its rendered formats such as HTML. Navigate to your Ceph repository and view a document in its native format. You may notice that it is generally as legible in a terminal as it is in its rendered HTML format. Additionally, you may also notice that diagrams in `ditaa` format also render reasonably well in text mode.

```
less doc/architecture.rst
```

Review the following style guides to maintain this consistency.

HEADINGS

1. **Document Titles:** Document titles use the = character overline and underline with a leading and trailing space on the title text line. See Document Title for details.
2. **Section Titles:** Section tiles use the = character underline with no leading or trailing spaces for text. Two carriage returns should precede a section title (unless an inline reference precedes it). See Sections for details.
3. **Subsection Titles:** Subsection titles use the _ character underline with no leading or trailing spaces for text. Two carriage returns should precede a subsection title (unless an inline reference precedes it).

TEXT BODY

As a general rule, we prefer text to wrap at column 80 so that it is legible in a command line interface without leading or trailing white space. Where possible, we prefer to maintain this convention with text, lists, literal text (exceptions allowed), tables, and `ditaa` graphics.

1. **Paragraphs**: Paragraphs have a leading and a trailing carriage return, and should be 80 characters wide or less so that the documentation can be read in native format in a command line terminal.
2. **Literal Text:** To create an example of literal text (e.g., command line usage), terminate the preceding paragraph with `::` or enter a carriage return to create an empty line after the preceding paragraph; then, enter `::` on a separate line followed by another empty line. Then, begin the literal text with tab indentation (preferred) or space indentation of 3 characters.
3. **Indented Text:** Indented text such as bullet points (e.g., `- some text`) may span multiple lines. The text of subsequent lines should begin at the same character position as the text of the indented text

(less numbers, bullets, etc.).

Indented text may include literal text examples. Whereas, text indentation should be done with spaces, literal text examples should be indented with tabs. This convention enables you to add an additional indented paragraph following a literal example by leaving a blank line and beginning the subsequent paragraph with space indentation.

4. **Numbered Lists:** Numbered lists should use autonumbering by starting a numbered indent with `#.` instead of the actual number so that numbered paragraphs can be repositioned without requiring manual renumbering.
5. **Code Examples:** Ceph supports the use of the `.. code-block::<language>` role, so that you can add highlighting to source examples. This is preferred for source code. However, use of this tag will cause autonumbering to restart at 1 if it is used as an example within a numbered list. See Showing code examples for details.

PARAGRAPH LEVEL MARKUP

The Ceph project uses paragraph level markup to highlight points.

1. **Tip:** Use the `.. tip::` directive to provide additional information that assists the reader or steers the reader away from trouble.
2. **Note**: Use the `.. note::` directive to highlight an important point.
3. **Important:** Use the `.. important::` directive to highlight important requirements or caveats (e.g., anything that could lead to data loss). Use this directive sparingly, because it renders in red.
4. **Version Added:** Use the `.. versionadded::` directive for new features or configuration settings so that users know the minimum release for using a feature.
5. **Version Changed:** Use the `.. versionchanged::` directive for changes in usage or configuration settings.
6. **Deprecated:** Use the `.. deprecated::` directive when CLI usage, a feature or a configuration setting is no longer preferred or will be discontinued.
7. **Topic:** Use the `.. topic::` directive to encapsulate text that is outside the main flow of the document. See the topic directive for additional details.

TOC AND HYPERLINKS

All documents must be linked from another document or a table of contents, otherwise you will receive a warning when building the documentation.

The Ceph project uses the `.. toctree::` directive. See The TOC tree for details. When rendering a TOC, consider specifying the `:maxdepth:` parameter so the rendered TOC is reasonably terse.

Document authors should prefer to use the `:ref:` syntax where a link target contains a specific unique identifier (e.g., `.. _unique-target-id:`), and a reference to the target specifically references the target (e.g., ``:ref:`unique-target-id` ``) so that if source files are moved or the information architecture changes, the links will still work. See Cross referencing arbitrary locations for details.

Ceph documentation also uses the backtick (accent grave) character followed by the link text, another backtick and an underscore. Sphinx allows you to incorporate the link destination inline; however, we prefer to use the use the `.. _Link Text: ../path` convention at the bottom of the document, because it improves the readability of the document in a command line interface.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

INSTALLATION (CEPH-DAEMON)

A new Ceph cluster is deployed by bootstrapping a cluster on a single node, and then adding additional nodes and daemons via the CLI or GUI dashboard.

GET CEPH-DAEMON

The `ceph-daemon` utility is used to bootstrap a new Ceph Cluster. You can get the utility by either installing a package provided by your Linux distribution:

```
sudo apt install -y ceph-daemon   # or,
sudo dnf install -y ceph-daemon   # or,
sudo yum install -y ceph-daemon
```

or by simply downloading the standalone script manually:

```
curl --silent --remote-name --location https://github.com/ceph/ceph
chmod +x ceph-daemon
sudo install -m 0755 ceph-daemon /usr/sbin    # optional!
```

BOOTSTRAP A NEW CLUSTER

To create a new cluster, you need to know:

- Which *IP address* to use for the cluster's first monitor. This is normally just the IP for the first cluster node. If there are multiple networks and interfaces, be sure to choose one that will be accessible by any hosts accessing the Ceph cluster.

To bootstrap the cluster,:

```
sudo ceph-daemon bootstrap --mon-ip *<mon-ip>*
```

This command does a few things:

- Creates a monitor and manager daemon for the new cluster on the local host. A minimal configuration file needed to communicate with the new cluster is written to `ceph.conf` in the local directory.
- A copy of the `client.admin` administrative (privileged!) secret key is written to `ceph.client.admin.keyring` in the local directory.
- Generates a new SSH key, and adds the public key to the local root user's `/root/.ssh/authorized_keys` file. A copy of the public key is written to `ceph.pub` in the local directory.

INTERACTING WITH THE CLUSTER

You can easily start up a container that has all of the Ceph packages installed to interact with your cluster:

```
sudo ceph-daemon shell --config ceph.conf --keyring ceph.keyring
```

The `--config` and `--keyring` arguments will bind those local files to the default locations in `/etc/ceph` inside the container to allow the `ceph` CLI utility to work without additional arguments. Inside the container, you can check the cluster status with:

```
ceph status
```

In order to interact with the Ceph cluster outside of a container, you need to install the Ceph client packages and install the configuration and privileged administrator key in a global location:

```
sudo apt install -y ceph-common   # or,
sudo dnf install -y ceph-common   # or,
sudo yum install -y ceph-common

sudo install -m 0644 ceph.conf /etc/ceph/ceph.conf
sudo install -m 0600 ceph.keyring /etc/ceph/ceph.keyring
```

ADDING HOSTS TO THE CLUSTER

For each new host you'd like to add to the cluster, you need to do two things:

1. Install the cluster's public SSH key in the new host's root user's `authorized_keys` file. For example,:

```
cat ceph.pub | ssh root@*newhost* tee -a /root/.ssh/authorized_
```

2. Tell Ceph that the new node is part of the cluster:

```
ceph orchestrator host add *newhost*
```

DEPLOYING ADDITIONAL MONITORS

Normally a Ceph cluster has at least three (or, preferably, five) monitor daemons spread across different hosts. Since we are deploying a monitor, we again need to specify what IP address it will use, either as a simple IP address or as a CIDR network name.

To deploy additional monitors,:

```
ceph orchestrator mon update *<new-num-monitors>* *<host1:network1>
```

For example, to deploy a second monitor on `newhost` using an IP address in network `10.1.2.0/24`,:

```
ceph orchestrator mon update 2 newhost:10.1.2.0/24
```

DEPLOYING OSDS

To add an OSD to the cluster, you need to know the device name for the block device (hard disk or SSD) that will be used. Then,:

```
ceph orchestrator osd create *<host>*:*<path-to-device>*
```

For example, to deploy an OSD on host *newhost*'s SSD,:

```
ceph orchestrator osd create newhost:/dev/disk/by-id/ata-WDC_WDS200
```

DEPLOYING MANAGER DAEMONS

It is a good idea to have at least one backup manager daemon. To deploy one or more new manager daemons,:

```
ceph orchestrator mgr update *<new-num-mgrs>* [*<host1>* ...]
```

DEPLOYING MDSS

In order to use the CephFS file system, one or more MDS daemons is needed.

TBD

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

INSTALLATION (CEPH-DEPLOY)

STEP 1: PREFLIGHT

A Ceph Client and a Ceph Node may require some basic configuration work prior to deploying a Ceph Storage Cluster. You can also avail yourself of help by getting involved in the Ceph community.

STEP 2: STORAGE CLUSTER

Once you have completed your preflight checklist, you should be able to begin deploying a Ceph Storage Cluster.

STEP 3: CEPH CLIENT(S)

Most Ceph users don’t store objects directly in the Ceph Storage Cluster. They typically use at least one of Ceph Block Devices, the Ceph File System, and Ceph Object Storage.

- Configuring the Ceph Object Gateway Instance

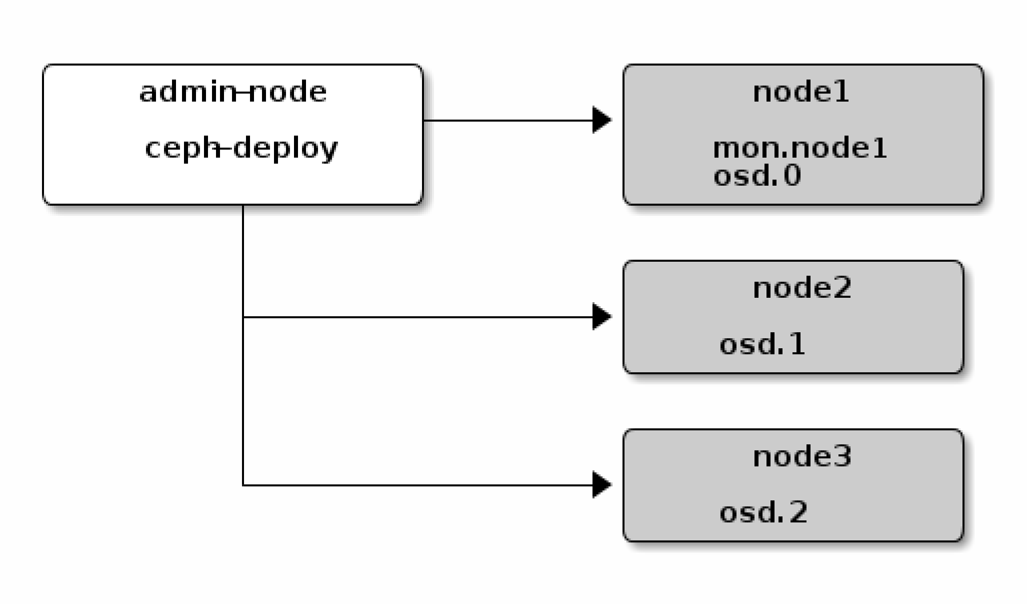
Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

PREFLIGHT CHECKLIST

The `ceph-deploy` tool operates out of a directory on an admin node. Any host with network connectivity and a modern python environment and ssh (such as Linux) should work.

In the descriptions below, Node refers to a single machine.



CEPH-DEPLOY SETUP

Add Ceph repositories to the `ceph-deploy` admin node. Then, install `ceph-deploy`.

DEBIAN/UBUNTU

For Debian and Ubuntu distributions, perform the following steps:

1. Add the release key:

```
wget -q -O- 'https://download.ceph.com/keys/release.asc' | sudo
```

2. Add the Ceph packages to your repository. Use the command below and replace `{ceph-stable-release}` with a stable Ceph release (e.g., `luminous`.) For example:

   ```
   echo deb https://download.ceph.com/debian-{ceph-stable-release}
   ```

3. Update your repository and install `ceph-deploy`:

   ```
   sudo apt update
   sudo apt install ceph-deploy
   ```

> **Note:** You can also use the EU mirror eu.ceph.com for downloading your packages by replacing `https://ceph.com/` by `http://eu.ceph.com/`

RHEL/CENTOS

For CentOS 7, perform the following steps:

1. On Red Hat Enterprise Linux 7, register the target machine with `subscription-manager`, verify your subscriptions, and enable the “Extras” repository for package dependencies. For example:

   ```
   sudo subscription-manager repos --enable=rhel-7-server-extras-r
   ```

2. Install and enable the Extra Packages for Enterprise Linux (EPEL) repository:

   ```
   sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-
   ```

   Please see the EPEL wiki page for more information.

3. Add the Ceph repository to your yum configuration file at `/etc/yum.repos.d/ceph.repo` with the following command. Replace `{ceph-stable-release}` with a stable Ceph release (e.g., `luminous`.) For example:

   ```
   cat << EOM > /etc/yum.repos.d/ceph.repo
   [ceph-noarch]
   name=Ceph noarch packages
   baseurl=https://download.ceph.com/rpm-{ceph-stable-release}/el7
   enabled=1
   gpgcheck=1
   type=rpm-md
   ```

```
gpgkey=https://download.ceph.com/keys/release.asc
EOM
```

4. Update your repository and install `ceph-deploy`:

```
sudo yum update
sudo yum install ceph-deploy
```

Note: You can also use the EU mirror eu.ceph.com for downloading your packages by replacing `https://ceph.com/` by `http://eu.ceph.com/`

OPENSUSE

The Ceph project does not currently publish release RPMs for openSUSE, but a stable version of Ceph is included in the default update repository, so installing it is just a matter of:

```
sudo zypper install ceph
sudo zypper install ceph-deploy
```

If the distro version is out-of-date, open a bug at https://bugzilla.opensuse.org/index.cgi and possibly try your luck with one of the following repositories:

1. Hammer:

```
https://software.opensuse.org/download.html?project=filesystems
```

2. Jewel:

```
https://software.opensuse.org/download.html?project=filesystems
```

CEPH NODE SETUP

The admin node must have password-less SSH access to Ceph nodes. When ceph-deploy logs in to a Ceph node as a user, that particular user must have passwordless `sudo` privileges.

INSTALL NTP

We recommend installing NTP on Ceph nodes (especially on Ceph Monitor nodes) to prevent issues arising from clock drift. See Clock for details.

On CentOS / RHEL, execute:

```
sudo yum install ntp ntpdate ntp-doc
```

On Debian / Ubuntu, execute:

```
sudo apt install ntp
```

Ensure that you enable the NTP service. Ensure that each Ceph Node uses the same NTP time server. See NTP for details.

INSTALL SSH SERVER

For **ALL** Ceph Nodes perform the following steps:

1. Install an SSH server (if necessary) on each Ceph Node:

   ```
   sudo apt install openssh-server
   ```

   or:

   ```
   sudo yum install openssh-server
   ```

2. Ensure the SSH server is running on **ALL** Ceph Nodes.

CREATE A CEPH DEPLOY USER

The `ceph-deploy` utility must login to a Ceph node as a user that has passwordless `sudo` privileges, because it needs to install software and configuration files without prompting for passwords.

Recent versions of `ceph-deploy` support a `--username` option so you can specify any user that has password-less `sudo` (including `root`, although this is **NOT** recommended). To use `ceph-deploy --username {username}`, the user you specify must have password-less SSH access to the Ceph node, as `ceph-deploy` will not prompt you for a password.

We recommend creating a specific user for `ceph-deploy` on **ALL** Ceph nodes in the cluster. Please do **NOT** use “ceph” as the user name. A uniform user name across the cluster may improve ease of use (not required), but you should avoid obvious user names, because hackers typically use them with brute force hacks (e.g., `root`, `admin`, `{productname}`). The following procedure, substituting `{username}` for the user name you define, describes how to create a user with passwordless `sudo`.

> **Note:** Starting with the Infernalis release, the “ceph” user name is reserved for the Ceph daemons. If the “ceph” user already exists on the Ceph nodes, removing the user must be done before attempting an upgrade.

1. Create a new user on each Ceph Node.

```
ssh user@ceph-server
sudo useradd -d /home/{username} -m {username}
sudo passwd {username}
```

2. For the new user you added to each Ceph node, ensure that the user has `sudo` privileges.

```
echo "{username} ALL = (root) NOPASSWD:ALL" | sudo tee /etc/sud
sudo chmod 0440 /etc/sudoers.d/{username}
```

ENABLE PASSWORD-LESS SSH

Since `ceph-deploy` will not prompt for a password, you must generate SSH keys on the admin node and distribute the public key to each Ceph node. `ceph-deploy` will attempt to generate the SSH keys for initial monitors.

1. Generate the SSH keys, but do not use `sudo` or the `root` user. Leave the passphrase empty:

```
ssh-keygen

Generating public/private key pair.
Enter file in which to save the key (/ceph-admin/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /ceph-admin/.ssh/id_rsa.
Your public key has been saved in /ceph-admin/.ssh/id_rsa.pub.
```

2. Copy the key to each Ceph Node, replacing `{username}` with the user name you created with Create a Ceph Deploy User.

```
ssh-copy-id {username}@node1
ssh-copy-id {username}@node2
ssh-copy-id {username}@node3
```

3. (Recommended) Modify the `~/.ssh/config` file of your `ceph-deploy` admin node so that `ceph-deploy` can log in to Ceph nodes as the user you created without requiring you to specify `--username {username}` each time you execute `ceph-deploy`. This has the added benefit of streamlining `ssh` and `scp` usage. Replace `{username}` with the user name you created:

   ```
   Host node1
      Hostname node1
      User {username}
   Host node2
      Hostname node2
      User {username}
   Host node3
      Hostname node3
      User {username}
   ```

ENABLE NETWORKING ON BOOTUP

Ceph OSDs peer with each other and report to Ceph Monitors over the network. If networking is `off` by default, the Ceph cluster cannot come online during bootup until you enable networking.

The default configuration on some distributions (e.g., CentOS) has the networking interface(s) off by default. Ensure that, during boot up, your network interface(s) turn(s) on so that your Ceph daemons can communicate over the network. For example, on Red Hat and CentOS, navigate to `/etc/sysconfig/network-scripts` and ensure that the `ifcfg-{iface}` file has `ONBOOT` set to `yes`.

ENSURE CONNECTIVITY

Ensure connectivity using `ping` with short hostnames (`hostname -s`). Address hostname resolution issues as necessary.

> **Note:** Hostnames should resolve to a network IP address, not to the loopback IP address (e.g., hostnames should resolve to an IP address other than `127.0.0.1`). If you use your admin node as a Ceph node, you should also ensure that it resolves to its hostname and IP address (i.e., not its loopback IP address).

OPEN REQUIRED PORTS

Ceph Monitors communicate using port `6789` by default. Ceph OSDs communicate in a port range of `6800:7300` by default. See the Network Configuration Reference for details. Ceph OSDs can use multiple network connections to communicate with clients, monitors, other OSDs for replication, and other OSDs for heartbeats.

On some distributions (e.g., RHEL), the default firewall configuration is fairly strict. You may need to adjust your firewall settings allow inbound requests so that clients in your network can communicate with daemons on your Ceph nodes.

For `firewalld` on RHEL 7, add the `ceph-mon` service for Ceph Monitor nodes and the `ceph` service for Ceph OSDs and MDSs to the public zone and ensure that you make the settings permanent so that they are enabled on reboot.

For example, on monitors:

```
sudo firewall-cmd --zone=public --add-service=ceph-mon --permanent
```

and on OSDs and MDSs:

```
sudo firewall-cmd --zone=public --add-service=ceph --permanent
```

Once you have finished configuring firewalld with the `--permanent` flag, you can make the changes live immediately without rebooting:

```
sudo firewall-cmd --reload
```

For `iptables`, add port `6789` for Ceph Monitors and ports `6800:7300` for Ceph OSDs. For example:

```
sudo iptables -A INPUT -i {iface} -p tcp -s {ip-address}/{netmask}
```

Once you have finished configuring `iptables`, ensure that you make the changes persistent on each node so that they will be in effect when your nodes reboot. For example:

```
/sbin/service iptables save
```

TTY

On CentOS and RHEL, you may receive an error while trying to execute `ceph-deploy` commands. If `requiretty` is set by default on your Ceph nodes, disable it by executing `sudo visudo` and locate the `Defaults requiretty` setting. Change it to `Defaults:ceph !requiretty` or comment it out to ensure that `ceph-deploy` can connect using the user you created with Create a Ceph Deploy User.

Note: If editing, `/etc/sudoers`, ensure that you use `sudo visudo` rather than a text editor.

SELINUX

On CentOS and RHEL, SELinux is set to `Enforcing` by default. To streamline your installation, we recommend setting SELinux to `Permissive` or disabling it entirely and ensuring that your installation and cluster are working properly before hardening your configuration. To set SELinux to `Permissive`, execute the following:

```
sudo setenforce 0
```

To configure SELinux persistently (recommended if SELinux is an issue), modify the configuration file at `/etc/selinux/config`.

PRIORITIES/PREFERENCES

Ensure that your package manager has priority/preferences packages installed and enabled. On CentOS, you may need to install EPEL. On RHEL, you may need to enable optional repositories.

```
sudo yum install yum-plugin-priorities
```

For example, on RHEL 7 server, execute the following to install `yum-plugin-priorities` and enable the `rhel-7-server-optional-rpms` repository:

```
sudo yum install yum-plugin-priorities --enablerepo=rhel-7-server-o
```

SUMMARY

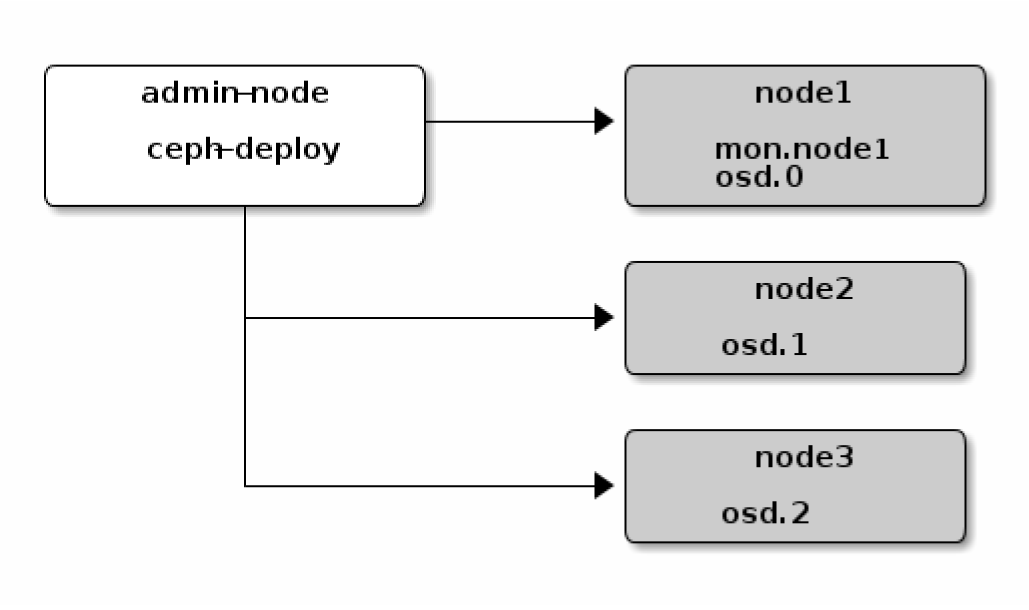
This completes the Quick Start Preflight. Proceed to the Storage Cluster Quick Start.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

STORAGE CLUSTER QUICK START

If you haven't completed your Preflight Checklist, do that first. This **Quick Start** sets up a Ceph Storage Cluster using `ceph-deploy` on your admin node. Create a three Ceph Node cluster so you can explore Ceph functionality.



As a first exercise, create a Ceph Storage Cluster with one Ceph Monitor and three Ceph OSD Daemons. Once the cluster reaches a `active + clean` state, expand it by adding a fourth Ceph OSD Daemon, and two more Ceph Monitors. For best results, create a directory on your admin node for maintaining the configuration files and keys that `ceph-deploy` generates for your cluster.

```
mkdir my-cluster
cd my-cluster
```

The `ceph-deploy` utility will output files to the current directory. Ensure you are in this directory when executing `ceph-deploy`.

Important: Do not call `ceph-deploy` with `sudo` or run it as `root` if you are logged in as a different user, because it will not issue `sudo` commands needed on the remote host.

STARTING OVER

If at any point you run into trouble and you want to start over, execute the following to purge the Ceph packages, and erase all its data and configuration:

```
ceph-deploy purge {ceph-node} [{ceph-node}]
ceph-deploy purgedata {ceph-node} [{ceph-node}]
ceph-deploy forgetkeys
rm ceph.*
```

If you execute `purge`, you must re-install Ceph. The last `rm` command removes any files that were written out by ceph-deploy locally during a previous installation.

CREATE A CLUSTER

On your admin node from the directory you created for holding your configuration details, perform the following steps using `ceph-deploy`.

1. Create the cluster.

   ```
   ceph-deploy new {initial-monitor-node(s)}
   ```

   Specify node(s) as hostname, fqdn or hostname:fqdn. For example:

   ```
   ceph-deploy new node1
   ```

   Check the output of `ceph-deploy` with `ls` and `cat` in the current directory. You should see a Ceph configuration file (`ceph.conf`), a monitor secret keyring (`ceph.mon.keyring`), and a log file for the new cluster. See ceph-deploy new -h for additional details.
2. If you have more than one network interface, add the `public network` setting under the `[global]` section of your Ceph configuration file. See the Network Configuration Reference for details.

   ```
   public network = {ip-address}/{bits}
   ```

   For example,:

   ```
   public network = 10.1.2.0/24
   ```

   to use IPs in the 10.1.2.0/24 (or 10.1.2.0/255.255.255.0) network.
3. If you are deploying in an IPv6 environment, add the following to `ceph.conf` in the local directory:

   ```
   echo ms bind ipv6 = true >> ceph.conf
   ```

4. Install Ceph packages.:

   ```
   ceph-deploy install {ceph-node} [...]
   ```

   For example:

   ```
   ceph-deploy install node1 node2 node3
   ```

   The `ceph-deploy` utility will install Ceph on each node.
5. Deploy the initial monitor(s) and gather the keys:

   ```
   ceph-deploy mon create-initial
   ```

   Once you complete the process, your local directory should have the following keyrings:

   - `ceph.client.admin.keyring`
   - `ceph.bootstrap-mgr.keyring`
   - `ceph.bootstrap-osd.keyring`
   - `ceph.bootstrap-mds.keyring`
   - `ceph.bootstrap-rgw.keyring`
   - `ceph.bootstrap-rbd.keyring`
   - `ceph.bootstrap-rbd-mirror.keyring`

   > **Note:** If this process fails with a message similar to “Unable to find /etc/ceph/ceph.client.admin.keyring”, please ensure that the IP listed for the monitor node in ceph.conf is the Public IP, not the Private IP.

6. Use `ceph-deploy` to copy the configuration file and admin key to your admin node and your Ceph Nodes so that you can use the `ceph` CLI without having to specify the monitor address and `ceph.client.admin.keyring` each time you execute a command.

   ```
   ceph-deploy admin {ceph-node(s)}
   ```

   For example:

   ```
   ceph-deploy admin node1 node2 node3
   ```

7. Deploy a manager daemon. (Required only for luminous+ builds):

```
ceph-deploy mgr create node1  *Required only for luminous+ buil
```

8. Add three OSDs. For the purposes of these instructions, we assume you have an unused disk in each node called `/dev/vdb`. *Be sure that the device is not currently in use and does not contain any important data.*

   ```
   ceph-deploy osd create --data {device} {ceph-node}
   ```

   For example:

   ```
   ceph-deploy osd create --data /dev/vdb node1
   ceph-deploy osd create --data /dev/vdb node2
   ceph-deploy osd create --data /dev/vdb node3
   ```

   Note: If you are creating an OSD on an LVM volume, the argument to `--data` *must* be `volume_group/lv_name`, rather than the path to the volume's block device.

9. Check your cluster's health.

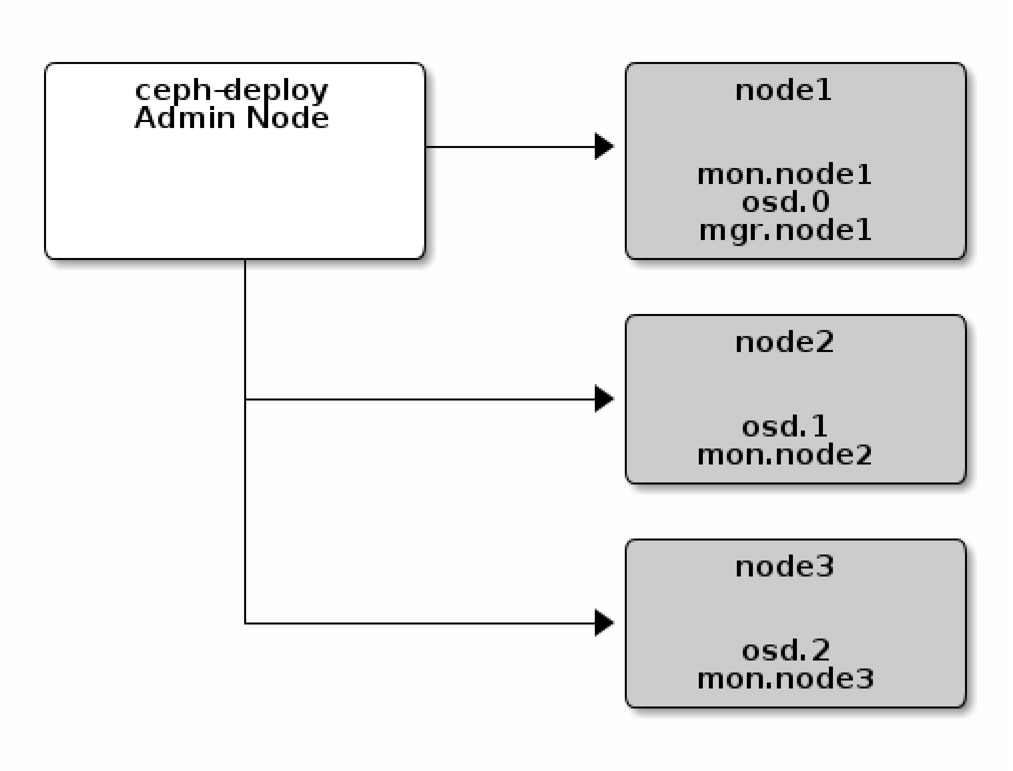
   ```
   ssh node1 sudo ceph health
   ```

   Your cluster should report `HEALTH_OK`. You can view a more complete cluster status with:

   ```
   ssh node1 sudo ceph -s
   ```

EXPANDING YOUR CLUSTER

Once you have a basic cluster up and running, the next step is to expand cluster. Then add a Ceph Monitor and Ceph Manager to `node2` and `node3` to improve reliability and availability.



ADDING MONITORS

A Ceph Storage Cluster requires at least one Ceph Monitor and Ceph Manager to run. For high availability, Ceph Storage Clusters typically run multiple Ceph Monitors so that the failure of a single Ceph Monitor will not bring down the Ceph Storage Cluster. Ceph uses the Paxos algorithm, which requires a majority of monitors (i.e., greater than *N/2* where *N* is the number of monitors) to form a quorum. Odd numbers of monitors tend to be better, although this is not required.

Add two Ceph Monitors to your cluster:

```
ceph-deploy mon add {ceph-nodes}
```

For example:

```
ceph-deploy mon add node2 node3
```

Once you have added your new Ceph Monitors, Ceph will begin synchronizing the monitors and form a quorum. You can check the quorum status by executing the following:

```
ceph quorum_status --format json-pretty
```

Tip: When you run Ceph with multiple monitors, you SHOULD install and configure NTP on each monitor host. Ensure that the monitors are NTP peers.

ADDING MANAGERS

The Ceph Manager daemons operate in an active/standby pattern. Deploying additional manager daemons ensures that if one daemon or host fails, another one can take over without interrupting service.

To deploy additional manager daemons:

```
ceph-deploy mgr create node2 node3
```

You should see the standby managers in the output from:

```
ssh node1 sudo ceph -s
```

ADD AN RGW INSTANCE

To use the Ceph Object Gateway component of Ceph, you must deploy an instance of RGW. Execute the following to create an new instance of RGW:

```
ceph-deploy rgw create {gateway-node}
```

For example:

```
ceph-deploy rgw create node1
```

By default, the RGW instance will listen on port 7480. This can be changed by editing ceph.conf on the node running the RGW as follows:

```
[client]
rgw frontends = civetweb port=80
```

To use an IPv6 address, use:

```
[client]
rgw frontends = civetweb port=[::]:80
```

STORING/RETRIEVING OBJECT DATA

To store object data in the Ceph Storage Cluster, a Ceph client must:

1. Set an object name
2. Specify a pool

The Ceph Client retrieves the latest cluster map and the CRUSH algorithm calculates how to map the object to a placement group, and then calculates how to assign the placement group to a Ceph OSD Daemon dynamically. To find the object location, all you need is the object name and the pool name. For example:

```
ceph osd map {poolname} {object-name}
```

Exercise: Locate an Object

As an exercise, lets create an object. Specify an object name, a path to a test file containing some object data and a pool name using the `rados put` command on the command line. For example:

```
echo {Test-data} > testfile.txt
ceph osd pool create mytest
rados put {object-name} {file-path} --pool=mytest
rados put test-object-1 testfile.txt --pool=mytest
```

To verify that the Ceph Storage Cluster stored the object, execute the following:

```
rados -p mytest ls
```

Now, identify the object location:

```
ceph osd map {pool-name} {object-name}
ceph osd map mytest test-object-1
```

Ceph should output the object's location. For example:

```
osdmap e537 pool 'mytest' (1) object 'test-object-1' -> pg 1.d174
```

To remove the test object, simply delete it using the `rados rm` command.

For example:

```
rados rm test-object-1 --pool=mytest
```

To delete the `mytest` pool:

```
ceph osd pool rm mytest
```

(For safety reasons you will need to supply additional arguments as prompted; deleting pools destroys data.)

As the cluster evolves, the object location may change dynamically. One benefit of Ceph’s dynamic rebalancing is that Ceph relieves you from having to perform data migration or balancing manually.

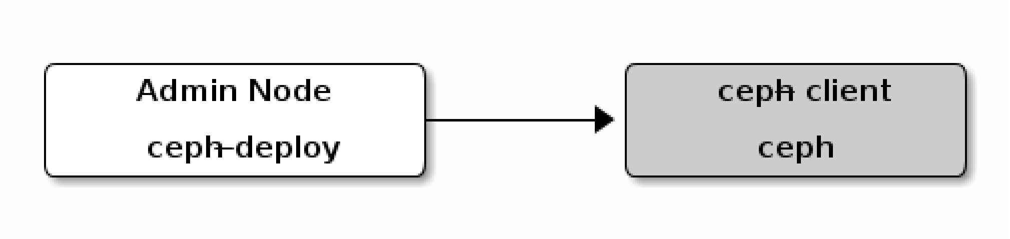
Notice: This document is for a development version of Ceph.

Edit on GitHub | Report a Documentation Bug

BLOCK DEVICE QUICK START

To use this guide, you must have executed the procedures in the Storage Cluster Quick Start guide first. Ensure your Ceph Storage Cluster is in an `active + clean` state before working with the Ceph Block Device.

Note: The Ceph Block Device is also known as RBD or RADOS Block Device.



You may use a virtual machine for your `ceph-client` node, but do not execute the following procedures on the same physical node as your Ceph Storage Cluster nodes (unless you use a VM). See FAQ for details.

INSTALL CEPH

1. Verify that you have an appropriate version of the Linux kernel. See OS Recommendations for details.

```
lsb_release -a
uname -r
```

2. On the admin node, use `ceph-deploy` to install Ceph on your `ceph-client` node.

```
ceph-deploy install ceph-client
```

3. On the admin node, use `ceph-deploy` to copy the Ceph configuration file and the `ceph.client.admin.keyring` to the `ceph-client`.

```
ceph-deploy admin ceph-client
```

The ceph-deploy utility copies the keyring to the /etc/ceph directory. Ensure that the keyring file has appropriate read permissions (e.g., sudo chmod +r /etc/ceph/ceph.client.admin.keyring).

CREATE A BLOCK DEVICE POOL

1. On the admin node, use the ceph tool to create a pool (we recommend the name 'rbd').
2. On the admin node, use the rbd tool to initialize the pool for use by RBD:

```
rbd pool init <pool-name>
```

CONFIGURE A BLOCK DEVICE

1. On the ceph-client node, create a block device image.

```
rbd create foo --size 4096 --image-feature layering [-m {mon-IP
```

2. On the ceph-client node, map the image to a block device.

```
sudo rbd map foo --name client.admin [-m {mon-IP}] [-k /path/to
```

3. Use the block device by creating a file system on the ceph-client node.

```
sudo mkfs.ext4 -m0 /dev/rbd/{pool-name}/foo

This may take a few moments.
```

4. Mount the file system on the ceph-client node.

```
sudo mkdir /mnt/ceph-block-device
sudo mount /dev/rbd/{pool-name}/foo /mnt/ceph-block-device
cd /mnt/ceph-block-device
```

5. Optionally configure the block device to be automatically mapped and mounted at boot (and unmounted/unmapped at shutdown) - see the rbdmap manpage.

See block devices for additional details.

Notice: This document is for a development version of Ceph.

Edit on GitHub | Report a Documentation Bug

CEPHFS QUICK START

To use the CephFS Quick Start guide, you must have executed the procedures in the Storage Cluster Quick Start guide first. Execute this quick start on the admin host.

PREREQUISITES

1. Verify that you have an appropriate version of the Linux kernel. See OS Recommendations for details.

```
lsb_release -a
uname -r
```

2. On the admin node, use `ceph-deploy` to install Ceph on your `ceph-client` node.

```
ceph-deploy install ceph-client
```

3. Optionally, if you want a FUSE-mounted file system, you would need to install `ceph-fuse` package as well.
4. Ensure that the Ceph Storage Cluster is running and in an `active + clean` state.

```
ceph -s [-m {monitor-ip-address}] [-k {path/to/ceph.client.admi
```

DEPLOY METADATA SERVER

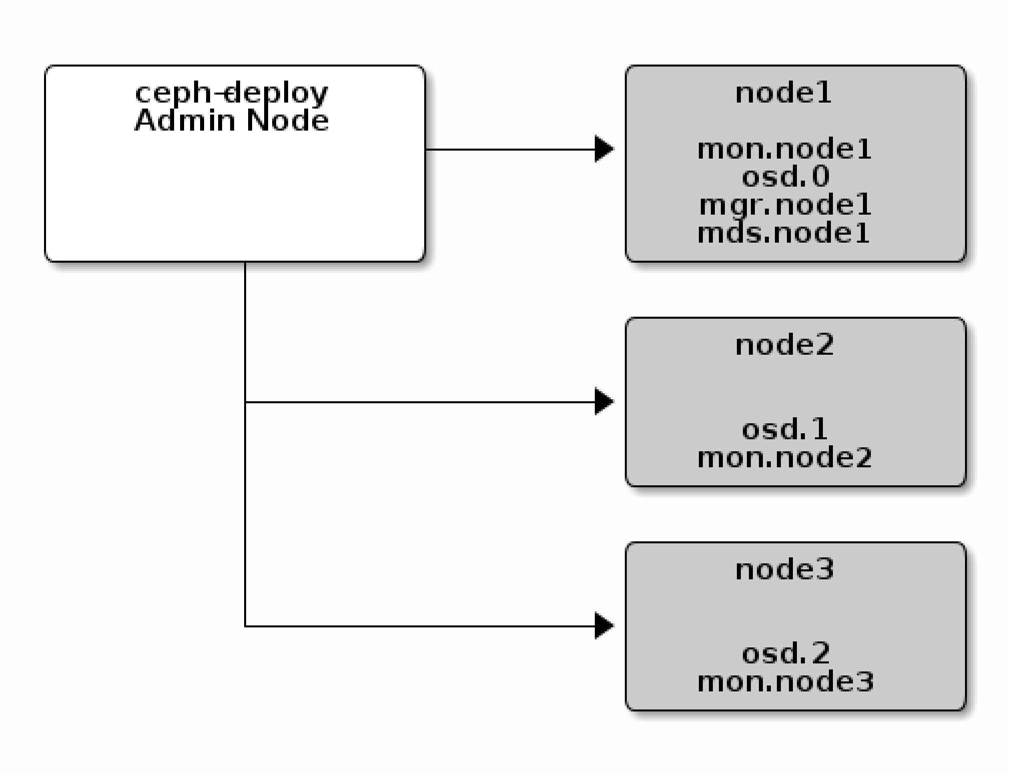
All metadata operations in CephFS happen via a metadata server, so you need at least one metadata server. Execute the following to create a metadata server:

```
ceph-deploy mds create {ceph-node}
```

For example:

```
ceph-deploy mds create node1
```

Now, your Ceph cluster would look like this:



CREATE A FILE SYSTEM

You have already created an MDS (Storage Cluster Quick Start) but it will not become active until you create some pools and a file system. See Create a Ceph file system.

```
ceph osd pool create cephfs_data 32
ceph osd pool create cephfs_meta 32
ceph fs new mycephfs cephfs_meta cephfs_data
```

Note: In case you have multiple Ceph applications and/or have multiple CephFSs on the same cluster, it would be easier to name your pools as <application>.<fs-name>.<pool-name>. In that case, the above pools would be named as cephfs.mycehfs.data and cephfs.mycehfs.meta.

QUICK WORD ABOUT POOLS AND PGS

REPLICATION NUMBER/POOL SIZE

Since the default replication number/size is 3, you'd need 3 OSDs to get `active+clean` for all PGs. Alternatively, you may change the replication number for the pool to match the number of OSDs:

```
ceph osd pool set cephfs_data size {number-of-osds}
ceph osd pool set cephfs_meta size {number-of-osds}
```

Usually, setting `pg_num` to 32 gives a perfectly healthy cluster. To pick appropriate value for `pg_num`, refer Placement Group. You can also use pg_autoscaler plugin instead. Introduced by Nautilus release, it can automatically increase/decrease value of `pg_num`; refer the Placement Group to find out more about it.

WHEN ALL OSDS ARE ON THE SAME NODE…

And, in case you have deployed all of the OSDs on the same node, you would need to create a new CRUSH rule to replicate data across OSDs and set the rule on the CephFS pools, since the default CRUSH rule is to replicate data across different nodes:

```
ceph osd crush rule create-replicated rule_foo default osd
ceph osd pool set cephfs_data crush_rule rule_foo
ceph osd pool set cephfs_meta crush_rule rule_foo
```

USING ERASURE CODED POOLS

You may also use Erasure Coded pools which can be more effecient and cost-saving since they allow stripping object data across OSDs and replicating these stripes with encoded redundancy information. The number of OSDs across which the data is stripped is *k* and number of replica is *m*. You'll need to pick up these values before creating CephFS pools. The following commands create a erasure code profile, creates a pool that'll use it and then enables it on the pool:

```
ceph osd erasure-code-profile set ec-42-profile k=4 m=2 crush-failu
ceph osd pool create cephfs_data_ec42 64 erasure ec-42-profile
ceph osd pool set cephfs_data_ec42 allow_ec_overwrites true
ceph fs add_data_pool mycephfs cephfs_data_ec42
```

You can also mark directories so that they are only stored on certain pools:

```
setfattr -n ceph.dir.layout -v pool=cephfs_data_ec42 /mnt/mycephfs/
```

This way you can choose the replication strategy for each directory on your Ceph file system.

Note: Erasure Coded pools can not be used for CephFS metadata pools.

Erasure coded pool were introduced in Firefly and could be used directly by CephFS Luminous onwards. Refer this article by Sage Weil to understand EC, it's background, limitations and other details in Ceph's context. Read more about Erasure Code here.

MOUNTING THE FILE SYSTEM

USING KERNEL DRIVER

The command to mount CephFS using kernel driver looks like this:

```
sudo mount -t ceph :{path-to-mounted} {mount-point} -o name={user-n
sudo mount -t ceph :/ /mnt/mycephfs -o name=admin   # usable versio
```

`{path-to-be-mounted}` is the path within CephFS that will be mounted, `{mount-point}` is the point in your file system upon which CephFS will be mounted and `{user-name}` is the name of CephX user that has the authorization to mount CephFS on the machine. Following command is the extended form, however these extra details are automatically figured out by by the mount.ceph helper program:

```
sudo mount -t ceph {ip-address-of-MON}:{port-number-of-MON}:{path-t
```

If you have multiple file systems on your cluster you would need to pass `mds_namespace={fs-name}` option to `-o` option to the `mount` command:

```
sudo mount -t ceph :/ /mnt/kcephfs2 -o name=admin,mds_namespace=myc
```

Refer mount.ceph man page and Mount CephFS using Kernel Driver to read more about this.

USING FUSE

To mount CephFS using FUSE (Filesystem in User Space) run:

```
sudo ceph-fuse /mnt/mycephfs
```

To mount a particular directory within CephFS you can use `-r`:

```
sudo ceph-fuse -r {path-to-be-mounted} /mnt/mycephfs
```

If you have multiple file systems on your cluster you would need to pass `--client_mds_namespace {fs-name}` to the `ceph-fuse` command:

```
sudo ceph-fuse /mnt/mycephfs2 --client_mds_namespace mycephfs2
```

Refer ceph-fuse man page and Mount CephFS using FUSE to read more about this.

Note: Mount the CephFS file system on the admin node, not the server node.

ADDITIONAL INFORMATION

See CephFS for additional information. See Troubleshooting if you encounter trouble.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

CEPH OBJECT GATEWAY QUICK START

As of *firefly* (v0.80), Ceph Storage dramatically simplifies installing and configuring a Ceph Object Gateway. The Gateway daemon embeds Civetweb, so you do not have to install a web server or configure FastCGI. Additionally, `ceph-deploy` can install the gateway package, generate a key, configure a data directory and create a gateway instance for you.

Tip: Civetweb uses port `7480` by default. You must either open port `7480`, or set the port to a preferred port (e.g., port `80`) in your Ceph configuration file.

To start a Ceph Object Gateway, follow the steps below:

INSTALLING CEPH OBJECT GATEWAY

1. Execute the pre-installation steps on your `client-node`. If you intend to use Civetweb's default port `7480`, you must open it using either `firewall-cmd` or `iptables`. See Preflight Checklist for more information.
2. From the working directory of your administration server, install the Ceph Object Gateway package on the `client-node` node. For example:

```
ceph-deploy install --rgw <client-node> [<client-node> ...]
```

CREATING THE CEPH OBJECT GATEWAY INSTANCE

From the working directory of your administration server, create an instance of the Ceph Object Gateway on the `client-node`. For example:

```
ceph-deploy rgw create <client-node>
```

Once the gateway is running, you should be able to access it on port `7480`. (e.g., `http://client-node:7480`).

CONFIGURING THE CEPH OBJECT GATEWAY INSTANCE

1. To change the default port (e.g., to port `80`), modify your Ceph configuration file. Add a section entitled `[client.rgw.<client-node>]`, replacing `<client-node>` with the short node name of your Ceph client node (i.e., `hostname -s`). For example, if your node name is `client-node`, add a section like this after the `[global]` section:

   ```
   [client.rgw.client-node]
   rgw_frontends = "civetweb port=80"
   ```

   Note: Ensure that you leave no whitespace between `port=<port-number>` in the `rgw_frontends` key/value pair.

   Important: If you intend to use port 80, make sure that the Apache server is not running otherwise it will conflict with Civetweb. We recommend to remove Apache in this case.

2. To make the new port setting take effect, restart the Ceph Object Gateway. On Red Hat Enterprise Linux 7 and Fedora, run the following command:

   ```
   sudo systemctl restart ceph-radosgw.service
   ```

   On Red Hat Enterprise Linux 6 and Ubuntu, run the following command:

   ```
   sudo service radosgw restart id=rgw.<short-hostname>
   ```

3. Finally, check to ensure that the port you selected is open on the node's firewall (e.g., port `80`). If it is not open, add the port and reload the firewall configuration. For example:

   ```
   sudo firewall-cmd --list-all
   sudo firewall-cmd --zone=public --add-port 80/tcp --permanent
   sudo firewall-cmd --reload
   ```

   See Preflight Checklist for more information on configuring firewall with `firewall-cmd` or `iptables`.

   You should be able to make an unauthenticated request, and receive a response. For example, a request with no parameters like this:

   ```
   http://<client-node>:80
   ```

   Should result in a response like this:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <ListAllMyBucketsResult xmlns="http://s3.amazonaws.com/doc/2006
   ```

```
  <Owner>
    <ID>anonymous</ID>
    <DisplayName></DisplayName>
  </Owner>
    <Buckets>
  </Buckets>
</ListAllMyBucketsResult>
```

See the Configuring Ceph Object Gateway guide for additional administration and API details.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

INSTALLATION (MANUAL)

GET SOFTWARE

There are several methods for getting Ceph software. The easiest and most common method is to get packages by adding repositories for use with package management tools such as the Advanced Package Tool (APT) or Yellowdog Updater, Modified (YUM). You may also retrieve pre-compiled packages from the Ceph repository. Finally, you can retrieve tarballs or clone the Ceph source code repository and build Ceph yourself.

- Get Packages
- Get Tarballs
- Clone Source
- Build Ceph
- Ceph Mirrors

INSTALL SOFTWARE

Once you have the Ceph software (or added repositories), installing the software is easy. To install packages on each Ceph Node in your cluster. You may use `ceph-deploy` to install Ceph for your storage cluster, or use package management tools. You should install Yum Priorities for RHEL/CentOS and other distributions that use Yum if you intend to install the Ceph Object Gateway or QEMU.

- Install ceph-deploy
- Install Ceph Storage Cluster
- Install Ceph Object Gateway
- Install Virtualization for Block

DEPLOY A CLUSTER MANUALLY

Once you have Ceph installed on your nodes, you can deploy a cluster manually. The manual procedure is primarily for exemplary purposes for those developing deployment scripts with Chef, Juju, Puppet, etc.

- Manual Deployment
  - Monitor Bootstrapping
  - Manager daemon configuration

UPGRADE SOFTWARE

As new versions of Ceph become available, you may upgrade your cluster to take advantage of new functionality. Read the upgrade documentation before you upgrade your cluster. Sometimes upgrading Ceph requires you to follow an upgrade sequence.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

GET PACKAGES

To install Ceph and other enabling software, you need to retrieve packages from the Ceph repository. Follow this guide to get packages; then, proceed to the Install Ceph Object Storage.

GETTING PACKAGES

There are two ways to get packages:

- **Add Repositories:** Adding repositories is the easiest way to get packages, because package management tools will retrieve the packages and all enabling software for you in most cases. However, to use this approach, each Ceph Node in your cluster must have internet access.
- **Download Packages Manually:** Downloading packages manually is a convenient way to install Ceph if your environment does not allow a Ceph Node to access the internet.

REQUIREMENTS

All Ceph deployments require Ceph packages (except for development). You should also add keys and recommended packages.

- **Keys: (Recommended)** Whether you add repositories or download packages manually, you should download keys to verify the packages. If you do not get the keys, you may encounter security warnings. See Add Keys for details.
- **Ceph: (Required)** All Ceph deployments require Ceph release packages, except for deployments that use development packages (development, QA, and bleeding edge deployments only). See Add Ceph for details.
- **Ceph Development: (Optional)** If you are developing for Ceph, testing Ceph development builds, or if you want features from the bleeding edge of Ceph development, you may get Ceph development packages. See Add Ceph Development for details.

If you intend to download packages manually, see Section Download Packages.

ADD KEYS

Add a key to your system's list of trusted keys to avoid a security warning. For major releases (e.g., `luminous`, `mimic`, `nautilus`) and development releases (`release-name-rc1`, `release-name-rc2`), use the `release.asc` key.

APT

To install the `release.asc` key, execute the following:

```
wget -q -O- 'https://download.ceph.com/keys/release.asc' | sudo apt
```

RPM

To install the `release.asc` key, execute the following:

```
sudo rpm --import 'https://download.ceph.com/keys/release.asc'
```

ADD CEPH

Release repositories use the `release.asc` key to verify packages. To install Ceph packages with the Advanced Package Tool (APT) or Yellowdog Updater, Modified (YUM), you must add Ceph repositories.

You may find releases for Debian/Ubuntu (installed with APT) at:

```
https://download.ceph.com/debian-{release-name}
```

You may find releases for CentOS/RHEL and others (installed with YUM) at:

```
https://download.ceph.com/rpm-{release-name}
```

The major releases of Ceph are summarized at: Ceph Releases (general)

Every second major release is considered Long Term Stable (LTS). Critical bugfixes are backported to LTS releases until their retirement. Since retired releases are no longer maintained, we recommend that users upgrade their clusters regularly - preferably to the latest LTS release.

> **Tip:** For non-US users: There might be a mirror close to you where to download Ceph from. For more information see: Ceph Mirrors.

DEBIAN PACKAGES

Add a Ceph package repository to your system’s list of APT sources. For newer versions of Debian/Ubuntu, call `lsb_release -sc` on the command line to get the short codename, and replace `{codename}` in the following command.

```
sudo apt-add-repository 'deb https://download.ceph.com/debian-lumin
```

For early Linux distributions, you may execute the following command:

```
echo deb https://download.ceph.com/debian-luminous/ $(lsb_release -
```

For earlier Ceph releases, replace `{release-name}` with the name with the name of the Ceph release. You may call `lsb_release -sc` on the command line to get the short codename, and replace `{codename}` in the following command.

```
sudo apt-add-repository 'deb https://download.ceph.com/debian-{rele
```

For older Linux distributions, replace `{release-name}` with the name of the release:

```
echo deb https://download.ceph.com/debian-{release-name}/ $(lsb_rel
```

For development release packages, add our package repository to your system's list of APT sources. See the testing Debian repository for a complete list of Debian and Ubuntu releases supported.

```
echo deb https://download.ceph.com/debian-testing/ $(lsb_release -s
```

> **Tip:** For non-US users: There might be a mirror close to you where to download Ceph from. For more information see: Ceph Mirrors.

RPM PACKAGES

RHEL

For major releases, you may add a Ceph entry to the `/etc/yum.repos.d` directory. Create a `ceph.repo` file. In the example below, replace `{ceph-release}` with a major release of Ceph (e.g., `luminous`, `mimic`, `nautilus`, etc.) and `{distro}` with your Linux distribution (e.g., `el7`, etc.). You may view https://download.ceph.com/rpm-{ceph-release}/ directory to see which distributions Ceph supports. Some Ceph packages (e.g., EPEL) must take priority over standard packages, so you must ensure that you set `priority=2`.

```
[ceph]
name=Ceph packages for $basearch
baseurl=https://download.ceph.com/rpm-{ceph-release}/{distro}/$base
enabled=1
priority=2
gpgcheck=1
gpgkey=https://download.ceph.com/keys/release.asc

[ceph-noarch]
name=Ceph noarch packages
baseurl=https://download.ceph.com/rpm-{ceph-release}/{distro}/noarc
enabled=1
priority=2
gpgcheck=1
gpgkey=https://download.ceph.com/keys/release.asc

[ceph-source]
name=Ceph source packages
baseurl=https://download.ceph.com/rpm-{ceph-release}/{distro}/SRPMS
enabled=0
priority=2
gpgcheck=1
gpgkey=https://download.ceph.com/keys/release.asc
```

For specific packages, you may retrieve them by downloading the release package by name. Our development process generates a new release of Ceph every 3-4 weeks. These packages are faster-moving than the major releases. Development packages have new features integrated quickly, while still undergoing several weeks of QA prior to release.

The repository package installs the repository details on your local system for use with `yum`. Replace `{distro}` with your Linux distribution, and `{release}` with the specific release of Ceph:

```
su -c 'rpm -Uvh https://download.ceph.com/rpms/{distro}/x86_64/ceph
```

You can download the RPMs directly from:

```
https://download.ceph.com/rpm-testing
```

Tip: For non-US users: There might be a mirror close to you where to download Ceph from. For more information see: Ceph Mirrors.

OPENSUSE LEAP 15.1

You need to add the Ceph package repository to your list of zypper sources. This can be done with the following command

```
zypper ar https://download.opensuse.org/repositories/filesystems:/c
```

OPENSUSE TUMBLEWEED

The newest major release of Ceph is already available through the normal Tumbleweed repositories. There’s no need to add another package repository manually.

ADD CEPH DEVELOPMENT

If you are developing Ceph and need to deploy and test specific Ceph branches, ensure that you remove repository entries for major releases first.

DEB PACKAGES

We automatically build Ubuntu packages for current development branches in the Ceph source code repository. These packages are intended for developers and QA only.

Add the package repository to your system’s list of APT sources, but replace `{BRANCH}` with the branch you’d like to use (e.g., wip-hack, master). See the shaman page for a complete list of distributions we build.

```
curl -L https://shaman.ceph.com/api/repos/ceph/{BRANCH}/latest/ubun
```

Note: If the repository is not ready an HTTP 504 will be returned

The use of `latest` in the url, means it will figure out which is the last commit that has been built. Alternatively, a specific sha1 can be specified. For Ubuntu Xenial and the master branch of Ceph, it would look like:

```
curl -L https://shaman.ceph.com/api/repos/ceph/master/53e772a45fdf2
```

Warning: Development repositories are no longer available after two weeks.

RPM PACKAGES

For current development branches, you may add a Ceph entry to the `/etc/yum.repos.d` directory. The the shaman page can be used to retrieve the full details of a repo file. It can be retrieved via an HTTP request, for example:

```
curl -L https://shaman.ceph.com/api/repos/ceph/{BRANCH}/latest/cent
```

The use of `latest` in the url, means it will figure out which is the last commit that has been built. Alternatively, a specific sha1 can be specified. For CentOS 7 and the master branch of Ceph, it would look like:

```
curl -L https://shaman.ceph.com/api/repos/ceph/master/53e772a45fdf2
```

Warning: Development repositories are no longer available after two weeks.

Note: If the repository is not ready an HTTP 504 will be returned

DOWNLOAD PACKAGES

If you are attempting to install behind a firewall in an environment without internet access, you must retrieve the packages (mirrored with all the necessary dependencies) before attempting an install.

DEBIAN PACKAGES

Ceph requires additional third party libraries.

- libaio1
- libsnappy1
- libcurl3
- curl
- libgoogle-perftools4
- google-perftools
- libleveldb1

The repository package installs the repository details on your local system for use with `apt`. Replace `{release}` with the latest Ceph release. Replace `{version}` with the latest Ceph version number. Replace `{distro}` with your Linux distribution codename. Replace `{arch}` with the CPU architecture.

```
wget -q https://download.ceph.com/debian-{release}/pool/main/c/ceph
```

RPM PACKAGES

Ceph requires additional additional third party libraries. To add the EPEL repository, execute the following:

```
sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-rele
```

Ceph requires the following packages:

- snappy
- leveldb
- gdisk
- python-argparse
- gperftools-libs

Packages are currently built for the RHEL/CentOS7 (`el7`) platforms. The repository package installs the repository details on your local system for use with `yum`. Replace `{distro}` with your distribution.

```
su -c 'rpm -Uvh https://download.ceph.com/rpm-luminous/{distro}/noa
```

For example, for CentOS 7 (`el7`):

```
su -c 'rpm -Uvh https://download.ceph.com/rpm-luminous/el7/noarch/c
```

You can download the RPMs directly from:

```
https://download.ceph.com/rpm-luminous
```

For earlier Ceph releases, replace `{release-name}` with the name with the name of the Ceph release. You may call `lsb_release -sc` on the command line to get the short codename.

```
su -c 'rpm -Uvh https://download.ceph.com/rpm-{release-name}/{distr
```

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

DOWNLOADING A CEPH RELEASE TARBALL

As Ceph development progresses, the Ceph team releases new versions of the source code. You may download source code tarballs for Ceph releases here:

Ceph Release Tarballs

Tip: For international users: There might be a mirror close to you where download Ceph from. For more information see: Ceph Mirrors.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

CLONING THE CEPH SOURCE CODE REPOSITORY

You may clone a Ceph branch of the Ceph source code by going to github Ceph Repository, selecting a branch (`master` by default), and clicking the **Download ZIP** button.

To clone the entire git repository, install and configure `git`.

INSTALL GIT

To install `git` on Debian/Ubuntu, execute:

```
sudo apt-get install git
```

To install `git` on CentOS/RHEL, execute:

```
sudo yum install git
```

You must also have a `github` account. If you do not have a `github` account, go to github.com and register. Follow the directions for setting up git at Set Up Git.

ADD SSH KEYS (OPTIONAL)

If you intend to commit code to Ceph or to clone using SSH (`git@github.com:ceph/ceph.git`), you must generate SSH keys for github.

Tip: If you only intend to clone the repository, you may use `git clone --recursive https://github.com/ceph/ceph.git` without generating SSH keys.

To generate SSH keys for `github`, execute:

```
ssh-keygen
```

Get the key to add to your `github` account (the following example assumes you used the default file path):

```
cat .ssh/id_rsa.pub
```

Copy the public key.

Go to your `github` account, click on “Account Settings” (i.e., the ‘tools’ icon); then, click “SSH Keys” on the left side navbar.

Click “Add SSH key” in the “SSH Keys” list, enter a name for the key, paste the key you generated, and press the “Add key” button.

CLONE THE SOURCE

To clone the Ceph source code repository, execute:

```
git clone --recursive https://github.com/ceph/ceph.git
```

Once `git clone` executes, you should have a full copy of the Ceph repository.

Tip: Make sure you maintain the latest copies of the submodules included in the repository. Running `git status` will tell you if the submodules are out of date.

```
cd ceph
git status
```

If your submodules are out of date, run:

```
git submodule update --force --init --recursive
```

CHOOSE A BRANCH

Once you clone the source code and submodules, your Ceph repository will be on the `master` branch by default, which is the unstable development branch. You may choose other branches too.

- `master`: The unstable development branch.
- `stable`: The bugfix branch.
- `next`: The release candidate branch.

```
git checkout master
```

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

BUILD CEPH

You can get Ceph software by retrieving Ceph source code and building it yourself. To build Ceph, you need to set up a development environment, compile Ceph, and then either install in user space or build packages and install the packages.

BUILD PREREQUISITES

Tip: Check this section to see if there are specific prerequisites for your Linux/Unix distribution.

Before you can build Ceph source code, you need to install several libraries and tools:

```
./install-deps.sh
```

Note: Some distributions that support Google’s memory profiler tool may use a different package name (e.g., `libgoogle-perftools4`).

BUILD CEPH

Ceph is built using cmake. To build Ceph, navigate to your cloned Ceph repository and execute the following:

```
cd ceph
./do_cmake.sh
cd build
make
```

Note: By default do_cmake.sh will build a debug version of ceph that may perform up to 5 times slower with certain workloads. Pass ‘-DCMAKE_BUILD_TYPE=RelWithDebInfo’ to do_cmake.sh if you would like to build a release version of the ceph executables instead.

Hyperthreading

You can use `make -j` to execute multiple jobs depending upon your system. For example, `make -j4` for a dual core processor may build faster.

See Installing a Build to install a build in user space.

BUILD CEPH PACKAGES

To build packages, you must clone the Ceph repository. You can create installation packages from the latest code using `dpkg-buildpackage` for Debian/Ubuntu or `rpmbuild` for the RPM Package Manager.

Tip: When building on a multi-core CPU, use the `-j` and the number of cores * 2. For example, use `-j4` for a dual-core processor to accelerate the build.

ADVANCED PACKAGE TOOL (APT)

To create `.deb` packages for Debian/Ubuntu, ensure that you have cloned the Ceph repository, installed the Build Prerequisites and installed `debhelper`:

```
sudo apt-get install debhelper
```

Once you have installed debhelper, you can build the packages:

```
sudo dpkg-buildpackage
```

For multi-processor CPUs use the `-j` option to accelerate the build.

RPM PACKAGE MANAGER

To create `.rpm` packages, ensure that you have cloned the Ceph repository, installed the Build Prerequisites and installed `rpm-build` and `rpmdevtools`:

```
yum install rpm-build rpmdevtools
```

Once you have installed the tools, setup an RPM compilation environment:

```
rpmdev-setuptree
```

Fetch the source tarball for the RPM compilation environment:

```
wget -P ~/rpmbuild/SOURCES/ https://download.ceph.com/tarballs/ceph
```

Or from the EU mirror:

```
wget -P ~/rpmbuild/SOURCES/ http://eu.ceph.com/tarballs/ceph-<versi
```

Extract the specfile:

```
tar --strip-components=1 -C ~/rpmbuild/SPECS/ --no-anchored -xvjf ~
```

Build the RPM packages:

```
rpmbuild -ba ~/rpmbuild/SPECS/ceph.spec
```

For multi-processor CPUs use the `-j` option to accelerate the build.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

CEPH MIRRORS

For improved user experience multiple mirrors for Ceph are available around the world.

These mirrors are kindly sponsored by various companies who want to support the Ceph project.

LOCATIONS

These mirrors are available on the following locations:

- **EU: Netherlands**: http://eu.ceph.com/
- **AU: Australia**: http://au.ceph.com/
- **SE: Sweden**: http://se.ceph.com/
- **DE: Germany**: http://de.ceph.com/
- **HK: Hong Kong**: http://hk.ceph.com/
- **FR: France**: http://fr.ceph.com/
- **UK: UK**: http://uk.ceph.com
- **US-East: US East Coast**: http://us-east.ceph.com/
- **US-Mid-West: Chicago**: http://mirrors.gigenet.com/ceph/
- **US-West: US West Coast**: http://us-west.ceph.com/
- **CN: China**: http://mirrors.ustc.edu.cn/ceph/

You can replace all download.ceph.com URLs with any of the mirrors, for example:

- http://download.ceph.com/tarballs/
- http://download.ceph.com/debian-hammer/
- http://download.ceph.com/rpm-hammer/

Change this to:

- http://eu.ceph.com/tarballs/
- http://eu.ceph.com/debian-hammer/
- http://eu.ceph.com/rpm-hammer/

MIRRORING

You can easily mirror Ceph yourself using a Bash script and rsync. A easy to use script can be found at Github.

When mirroring Ceph, please keep the following guidelines in mind:

- Choose a mirror close to you
- Do not sync in a interval shorter than 3 hours
- Avoid syncing at minute 0 of the hour, use something between 0 and 59

BECOMING A MIRROR

If you want to provide a public mirror for other users of Ceph you can opt to become a official mirror.

To make sure all mirrors meet the same standards some requirements have been set for all mirrors. These can be found on Github.

If you want to apply for an official mirror, please contact the ceph-users mailinglist.

Notice: This document is for a development version of Ceph.

Edit on GitHub | **Report a Documentation Bug**

INSTALL CEPH DEPLOY

The `ceph-deploy` tool enables you to set up and tear down Ceph clusters for development, testing and proof-of-concept projects.

APT

To install `ceph-deploy` with `apt`, execute the following:

```
sudo apt-get update && sudo apt-get install ceph-deploy
```

RPM

To install `ceph-deploy` with `yum`, execute the following:

```
sudo yum install ceph-deploy
```