Ceph Storage for Oracle® Linux Release 2.0

Release Notes

E66514-11
January 2019

Oracle Legal Notices

Table of Contents

Preface

This document contains information about Ceph Storage for Oracle Linux Release 2.0. It describes the differences from the upstream version, includes notes on installing and configuring Ceph, and provides a statement of what is supported.

Document generated on: 2019-01-17 (revision: 6938)

Audience

This document is written for developers who want to use Ceph with Oracle Linux 7. It is assumed that readers have a general understanding of the Linux operating system.

Related Documents

The latest version of this document and other documentation for this product are available at:

https://www.oracle.com/technetwork/server-storage/linux/documentation/index.html

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Chapter 1 Release Notes

1.1 About Ceph Storage for Oracle Linux Release 2.0

Ceph Storage for Oracle Linux Release 2.0 presents a uniform view of object and block storage from a cluster of multiple physical and logical commodity-hardware storage devices. Ceph can provide fault tolerance and enhance I/O performance by replicating and striping data across the storage devices in a Storage Cluster. Ceph's monitoring and self-repair features minimize administration overhead. You can configure a Storage Cluster on non-identical hardware from different manufacturers.

Ceph Storage for Oracle Linux Release 2.0 is based on the Ceph Community Jewel release (v10.2.2). Differences between Oracle versions of the software and upstream releases are limited to Oracle specific fixes and patches for specific bugs.

Availability

Ceph Storage for Oracle Linux Release 2.0 is available for Oracle Linux 7 (x86_64) running the Unbreakable Enterprise Kernel Release 4. A minimum of Oracle Linux 7 Update 2 (x86_64) and Unbreakable Enterprise Kernel Release 4 Update 2 is required.

Note

The source RPMs for Ceph are available from Oracle Yum at https://yum.oracle.com.

Supported Features and Technology Previews

The supported features include the Object Store, Block Device, Storage Cluster, Simple Ceph Object Gateway and Multisite Ceph Object Gateway components.

The following features are included as technology preview:

- Ceph Filesystem (Ceph FS)

Further Information

For a quick-start guide to using Ceph, see http://ceph.com/docs/master/start/quick-ceph-deploy/.

For more information about Ceph, go to http://ceph.com/.

1.2 Enabling Access to the Ceph Packages

The `ceph-deploy` package is available on Oracle Yum in the `ol7_ceph` channel or on ULN in the `ol7_x86_64_ceph` channel, however dependencies are scattered across some other channels as well and these must be enabled on each system where the Storage Cluster is deployed. On the Oracle Linux yum server, you must also enable the `ol7_addons`, `ol7_latest` and `ol7_optional_latest` channels. Alternatively, if you are using ULN, you must enable the `ol7_x86_64_addons`, `ol7_x86_64_latest` and `ol7_x86_64_optional_latest` channels.

Caution

Do not enable access to the `latest` repositories on both the Oracle Linux yum server and ULN.

If you are registered to use ULN, use the ULN web interface to subscribe the system to the appropriate channels:

1. Log in to https://linux.oracle.com with your ULN user name and password.
2. On the Systems tab, click the link named for the system in the list of registered machines.
3. On the System Details page, click **Manage Subscriptions**.
4. On the System Summary page, select each required channel from the list of available channels and click the right arrow to move the channel to the list of subscribed channels.

   Subscribe the system to the `ol7_x86_64_ceph`, `ol7_x86_64_addons`, `ol7_x86_64_latest`, and `ol7_x86_64_optional_latest` channels.
5. Click **Save Subscriptions**.

To enable the required channels using the Oracle Linux yum server,

ensure that your system is up to date and that you have transitioned to use the modular yum repository configuration by installing the `oraclelinux-release-el7` package and running the `/usr/bin/ol_yum_configure.sh` script.

```
# yum install oraclelinux-release-el7
# /usr/bin/ol_yum_configure.sh
```

Install the `oracle-ceph-release-el7` release package to install appropriate yum repository configuration.

```
# yum install oracle-ceph-release-el7
```

Enable the following repositories:

- `ol7_ceph`
- `ol7_addons`
- `ol7_latest`
- `ol7_optional_latest`

Use the `yum-config-manager` tool to update your yum configuration:

```
# yum-config-manager --enable ol7_ceph ol7_latest ol7_optional_latest ol7_addons
```

You can now prepare the Storage Cluster nodes for Ceph installation. See Section 1.3, “Installing and Configuring a Ceph Storage Cluster”.

1.3 Installing and Configuring a Ceph Storage Cluster

A Ceph Storage Cluster consists of several systems, known as nodes, each running the Ceph OSD (Object Storage Device) daemon. The Ceph storage cluster must also run the Ceph Monitor daemon on one or more nodes and may also run an optional Ceph Object Gateway on one or more nodes. A node is selected as an administration node from which commands can be run to control the cluster. Typically the administration node is also used as the deployment node, from which other systems can automatically be set up and configured as additional nodes within the cluster.

Note

For data integrity, a Storage Cluster should contain two or more nodes for storing copies of an object.

For high availability, a Storage Cluster should contain three or more nodes that store copies of an object.

In the example used in the following steps, the administration and deployment node is `ceph-node1.example.com` (192.168.1.51).

1.3.1 Preparing the Storage Cluster Nodes Before Installing Ceph

There are some basic requirements for each Oracle Linux system that you intend to use as a Storage Cluster node. These include the following items, for which some preparatory work may be required before you can begin your deployment:

1. Time must be accurate and synchronized across the nodes within the storage cluster. This is achieved by installing and configuring NTP on each system that you wish to run as a node in the cluster. If the NTP service if not already configured, install and start it. See the Oracle Linux 7 Administrator's Guide for more information on configuring NTP.

   

   Note

   Use the `hwclock --show` command to ensure that all nodes agree on the time. By default, the Ceph monitors report `health HEALTH_WARN clock skew detected on mon` errors if the clocks on the nodes differ by more than 50 milliseconds.

2. Cluster network communications must be able to take place between nodes within the cluster. If firewall software is running on any of the nodes, it must either be disabled or, preferably, configured to facilitate network traffic on the required ports.

   To stop and disable the firewall daemon on Oracle Linux 7, you can do the following:

   ```
   # systemctl stop firewalld
   # systemctl disable firewalld
   ```

   Preferably, leave the firewall running and configure the following rules:

   a. Allow TCP traffic on port 6789 to enable the Ceph Monitor:

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

   b. Allow TCP traffic for ports 6800 to 7300 to enable the traffic for the Ceph OSD daemon:

      ```
       # firewall-cmd --zone=public --add-port=6800-7300/tcp --permanent
      ```

   c. Allow TCP traffic on port 7480 to enable the Ceph Object Gateway:

      ```
      # firewall-cmd --zone=public --add-port=7480/tcp --permanent
      ```

   d. After modifying firewall rules, restart the firewall daemon service:

      ```
      # systemctl restart firewalld.service
      ```

3. Cluster nodes must be able to resolve the fully qualified domain name for each node within the cluster. You may either use DNS for this purpose, or provide entries within `/etc/hosts` for each system. If you select to rely on DNS, it must have sufficient redundancy to ensure that the cluster is able to perform name resolution at any time. If you select to edit `/etc/hosts`, add entries for the IP address and host name of all of the nodes in the Storage Cluster, for example:

```
192.168.1.51    ceph-node1.example.com ceph-node1
192.168.1.52    ceph-node2.example.com ceph-node2
192.168.1.53    ceph-node3.example.com ceph-node3
192.168.1.54    ceph-node4.example.com ceph-node4
```

Note

Although you can use DNS to configure host name to IP address mapping, Oracle recommends that you also configure `/etc/hosts` in case the DNS service becomes unavailable.

4. The Ceph Storage Cluster deployment node must be able to connect to each prospective node in the cluster over SSH, to facilitate deployment. To do this, you must generate an SSH key on the deployment node and copy the public key to each of the other nodes in the Storage Cluster.

   a. On the deployment node, generate the SSH key, specifying an empty passphrase:

   ```
   # ssh-keygen
   ```

   b. From the deployment node, copy the key to the other nodes in the Storage Cluster, for example:

   ```
   # ssh-copy-id root@ceph-node2
   # ssh-copy-id root@ceph-node3
   # ssh-copy-id root@ceph-node4
   ```

5. To prevent errors when running `ceph-deploy` as a user with passwordless `sudo` privileges, use `visudo` to comment out the `Defaults requiretty` setting in `/etc/sudoers` or change it to `Defaults:ceph !requiretty`.

You can now install and configure the Storage Cluster deployment node, which is usually the same system as the administration node. See Section 1.3.2, “Installing and Configuring Ceph on the Storage Cluster Deployment Node”.

1.3.2 Installing and Configuring Ceph on the Storage Cluster Deployment Node

Note

In the example used in the following steps, the deployment node is `ceph-node1.example.com` (192.168.1.51), which is the same as the administration node.

Perform the following steps on the deployment node:

1. Install the `ceph-deploy` package.

```
# yum install ceph-deploy
```

2. Create a Ceph configuration directory for the Storage Cluster and change to this directory, for example:

```
# mkdir /var/mydom_ceph
```

```
# cd /var/mydom_ceph
```

Note

This is the working configuration directory used by the Ceph deployment node to roll out configuration changes to the cluster and to client and gateway nodes. If you need to make changes to the Ceph configuration files in future, the changes should be made in this directory and then use the `ceph-deploy config push` command to update the configuration for other nodes in the cluster.

3. Use the `ceph-deploy` command to define the members of the Storage Cluster, for example:

```
# ceph-deploy --cluster mydom new ceph-node{1..4}
```

Tip

In this example we use the bash shell shorthand to iterate through adding nodes named ceph-node1, ceph-node2, ceph-node3 and ceph-node4. You may equally specify these hostnames manually as a space separated list. You may see this, and similar, notation used throughout this document.

Note

If you do not intend to run more than one Storage Cluster on the same hardware, you do not need to specify a cluster name using the `--cluster` option.

4. Edit `/var/mydom_ceph/ceph.conf` and set the default number of replicas, for example:

```
osd pool default size = 2
```

5. Edit `/var/mydom_ceph/ceph.conf` and add the workaround for the issue described in Section 1.8.3, “RBD kernel module fails to map an image to a block device”:

```
rbd default features = 3
```

You can now install Ceph on the remaining Storage Cluster nodes. See Section 1.3.3, “Installing and Configuring Ceph on Participating Storage Cluster Nodes”.

1.3.3 Installing and Configuring Ceph on Participating Storage Cluster Nodes

Having installed and configured the Ceph deployment node, you can use this node to install Ceph on the other nodes participating in the Storage Cluster.

To install Ceph on all the Storage Cluster nodes, run the following command on the deployment node:

```
# ceph-deploy install ceph-node{1..4}
```

To configure the Storage Cluster, perform the following steps on the administration node:

1. Initialize Ceph monitoring and deploy a Ceph Monitor on one or more nodes in the Storage Cluster, for example:

```
# ceph-deploy mon create-initial
# ceph-deploy mon create ceph-node{2,3,4}
```

Note

For high availability, Oracle recommends that you configure at least three nodes as Ceph Monitors.

2. Gather the monitor keys and the OSD and MDS bootstrap keyrings from one of the Ceph Monitors, for example:

```
# ceph-deploy gatherkeys ceph-node3
```

3. Use the following command to prepare the back-end storage devices for each node in the Storage Cluster:

```
# ceph-deploy osd create --zap-disk --fs-type fstype node:device
```

Note

This commands deletes all data on the specified device and alters partitioning on the device.

Replace *node* with the node name or hostname where the disk is located. Replace *device* with the device name for the disk as reported when you run `lsblk` on the host where the disk is located. The supported file system types (*fstype*) are `btrfs` and `xfs`. This command repartitions the disk to usually create two partitions, one to contain the data and the other to contain the journal.

For example, prepare a btrfs file system as the back-end storage device on `/dev/sdb` for a node in the Storage Cluster:

```
# ceph-deploy osd create --zap-disk --fs-type btrfs ceph-node1:sdb
```

4. Use the following commands to check the health and status of the Storage Cluster:

```
# ceph health
# ceph status
```

It usually takes several minutes for the Storage Cluster to stabilize before its health is shown as `HEALTH_OK`. You can also check the Cluster quorum status to get an indication on the quorum status of the cluster monitors:

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

Refer to the upstream Ceph documentation for help troubleshooting any issues with the health or status of your Storage Cluster.

1.4 Upgrade

Limited support for offline upgrades from Ceph Storage for Oracle Linux Release 1.0 to the current release is provided. Some additional configuration specific to your deployment may be required when you have completed the upgrade.

Where upgrade is required it is recommended that upgrades of components are performed in the following order:

1. Ceph Deploy Package
2. Ceph Monitors

3. Ceph OSD Daemons
4. Ceph Metadata Servers
5. Ceph Object Gateways

Oracle recommends that all daemons of a specific type are upgraded together to ensure that they are all on the same release, and that all of the components within a cluster are upgraded before you attempt to configure or use any new functionality in the current release.

The following instructions provide a brief outline of some of the common steps required to perform an upgrade for a storage cluster. Remember that the cluster must go offline during the upgrade.

1. To begin the upgrade process, the Yum configuration on all systems that are part of the Ceph Storage Cluster must be updated to provide access to the appropriate yum repositories and channels as described in Section 1.2, “Enabling Access to the Ceph Packages”.
2. Upgrade the Ceph Deploy package on the deployment node within your environment:

```
# yum upgrade ceph-deploy
```

3. Stop any running Ceph services on *each* of the different nodes within the environment:

```
# /etc/init.d/ceph stop mon
# /etc/init.d/ceph stop osd
# /etc/init.d/ceph stop mds
```

   Check the status of the cluster and ensure that it is not running:

```
# ceph status
```

4. Use Ceph deploy to install the package updates on each node in the storage cluster:

```
# ceph-deploy install ceph-node{1..4}
```

Note

The upstream documentation mentions the `--release` switch, which is meant to allow you to control which release you are updating to, however this switch does not have any effect when used in an Oracle Linux environment and the packages are simply installed to the latest version via Yum.

5. Add the following lines to the end of `/etc/ceph/ceph.conf` on *each* node:

```
rbd default feature = 3
setuser match path = /var/lib/ceph/$type/$cluster-$id
```

6. On each node, check that the `ceph` user and group have ownership of the directories used for Ceph:

```
# chown -R ceph:ceph /var/lib/ceph
# chown -R ceph:ceph /etc/ceph
```

7. On each Ceph monitor node, you must manually enable the systemd service for the Ceph monitor service. For example:

```
# systemctl enable ceph-mon@ceph-node1.service
```

   Start the service once you have enabled it and check its status to make sure that it is running properly:

```
# systemctl start ceph-mon@ceph-node1.service
```

```
# systemctl status ceph-mon@ceph-node1.service
```

8. From the administration node in the cluster, set the `noout` option to prevent the CRUSH algorithm from attempting to rebalance the cluster during the upgrade:

   ```
   # ceph osd set noout
   ```

   Also mark all OSDs as `down` within the cluster:

   ```
   # ceph osd down seq 0 1000
   ```

9. On each node that runs the Ceph OSD daemon, enable the systemd service, restart the service and check its status to make sure that it is running properly:

   ```
   # systemctl enable ceph-osd@0.service
   # systemctl restart ceph-osd@0.service
   # systemctl status ceph-osd@0.service
   ```

10. Reboot each Ceph OSD node once the `ceph-osd` services have been enabled.

11. From an admin node within the cluster, often the same as the deployment node, run the following command to tune the OSD instances:

    ```
    # ceph osd crush tunables default
    ```

    Updating the OSD tunable profile to the default setting requires that any Ceph client connecting to Ceph is up to date as well. Older Ceph clients designed to connect to previous releases may struggle to interface with a tuned Ceph deployment.

12. Check the cluster status with the following command:

    ```
    # ceph status
    ```

    If the health of the cluster is not set to `HEALTH_OK`, re-enable the CRUSH algorithm's ability to balance the cluster by unsetting the `noout` option:

    ```
    # ceph osd unset noout
    ```

13. On each node running the Ceph MDS daemon, enable the systemd service and restart the service:

    ```
    # systemctl enable ceph-mds@ceph-node1.service
    # systemctl start ceph-mds@ceph-node1.service
    ```

14. Comment out or remove the `setuser match path` parameter in `/etc/ceph/ceph.conf` on *each* node:

    ```
    # setuser match path = /var/lib/ceph/$type/$cluster-$id
    ```

    This parameter is only required during the upgrade. If it remains enabled, it can cause problems when the Ceph Object Gateway service, `radosgw`, is started.

15. On each node running the Ceph Object Gateway service, perform the following steps to complete the upgrade:

    a. Stop any running instances of the legacy Ceph Object Gateway service:

       ```
       # /etc/init.d/ceph-radosgw stop
       ```

    b. Stop the running Apache instance and disable it, so that it does not start at boot:

```
# systemctl stop httpd
# systemctl disable httpd
```

c. Edit the existing Ceph configuration for the Ceph Object Gateway by opening `/etc/ceph.conf` in an editor.

   An existing gateway configuration entry for a previous release should be similar to the following:

```
[client.radosgw.gateway]
host = ceph-node4
keyring = /etc/ceph/ceph.client.radosgw.keyring
rgw socket path = /var/run/ceph/ceph.radosgw.gateway.fastcgi.sock
log file = /var/log/radosgw/client.radosgw.gateway.log
rgw print continue = false
```

   Modify this entry to comment out the `rgw socket path` and `rgw print continue` parameters and to add an entry for the `rgw frontends` parameter that defines the port that the Civetweb web server should use. For example:

```
[client.radosgw.gateway]
host = ceph-node4
keyring = /etc/ceph/ceph.client.radosgw.keyring
# rgw socket path = /var/run/ceph/ceph.radosgw.gateway.fastcgi.sock
log file = /var/log/radosgw/client.radosgw.gateway.log
# rgw print continue = false
rgw_frontends = civetweb port=80
```

Tip

Take note of the configuration entry name. In the example, this is `client.radosgw.gateway`. You use this to specify the appropriate systemd service that should be enabled and run for this configuration. In this case, gateway name is `gateway`.

d. Enable and restart the systemd service and target for the gateway configuration. For example:

```
# systemctl enable ceph-radosgw@radosgw.gateway.service
# systemctl enable ceph-radosgw.target
# systemctl restart ceph-radosgw@radosgw.gateway.service
```

Important

Make sure that you specify the correct gateway name for the service. This should match the name in the configuration. In our example the gateway name is *`gateway`*.

e. Check the running status of the gateway service to make sure that the gateway is running and there are no errors:

```
# systemctl status ceph-radosgw@radosgw.gateway.service
```

f. Check that a client is able to access the gateway and list any existing buckets. See Section 1.6.1, “Simple Ceph Object Gateway” for an example test script.

1.5 Setting up and using a Ceph Block Device

To set up a Ceph Block Device, you must first install a Ceph Client on the system where you intend to use the block device. This is usually done from the administration and deployment nodes of your Ceph Storage Cluster.

1.5.1 Installing the Ceph Client

In this example, *`ceph-client`* is the hostname of the client system that is configured to access the Storage Cluster.

1. On the administration node of the Storage Cluster, copy the SSH key to the Ceph Client system, for example:

```
# ssh-copy-id root@ceph-client
```

Note

This example assumes that you have configured entries for the Ceph Client system in DNS and/or in `/etc/hosts`.

2. On the deployment node (which is usually the same as the administration node), use `ceph-deploy` to install Ceph on the Ceph Client system, for example:

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

3. On the administration node, copy the Ceph configuration file and the Ceph keyring to the Ceph Client system, for example:

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

1.5.2 Configuring a Block Device on a Ceph Client

Note

Ensure that the Storage Cluster is active and healthy before configuring a block device.

To configure a Block Device on a Ceph Client:

1. Create a storage pool for the block device within the OSD using the following command on the Ceph Client system:

```
# ceph osd pool create datastore 150 150
```

2. Use the `rbd` command to create a Block Device image in the pool, for example:

```
# rbd create --size 4096 --pool datastore vol01
```

This example creates a 4096 MB volume named `vol01` in the `datastore` pool.

Note

If you do not specify a storage pool, `rbd` uses the default `rbd` pool:

```
# rbd create --size 4096 vol01
```

3. Use the `rbd` command to map the image to a Block Device, for example:

```
# rbd map vol01 --pool datastore
```

Ceph creates the Block Device under `/dev/rbd/pool/volume`.

Note that the rbd kernel module is not loaded until you run this command. You can check that it is loaded after running the command by doing the following:

```
# lsmod|grep rbd
 rbd      73304   1
 libceph  235751  2  rbd,ceph
```

The `rbd ls` command lists the images that you have mapped for a storage pool, for example:

```
# rbd ls -p datastore
vol01
```

4. You can make a file system on the Block Device and mount this file system on a suitable mount point on the Ceph Client node, for example:

```
# mkfs.ext4 -m0 /dev/rbd/datastore/vol01
# mkdir /var/vol01
# mount /dev/rbd/datastore/vol01 /var/vol01
```

1.5.2.1 Removing a Block Device and its Storage Pool

To remove a Block Device and the Storage Pool where it is hosted:

1. Unmount any file system that is using the Block Device, for example:

```
# umount /var/vol01
```

2. Unmap the Block Device from its image, for example:

```
# rbd unmap /dev/rbd/datastore/vol01
```

3. Remove the Block Device image, for example:

```
# rbd rm vol01 -p datastore
```

4. Remove the Storage Pool, for example:

```
# ceph osd pool delete datastore datastore --yes-i-really-really-mean-it
```

1.6 Ceph Object Gateway

A Ceph Object Gateway provides a REST interface to the Ceph Storage Cluster to facilitate Amazon S3 and OpenStack Swift client access. The Ceph Object Gateway is described in more detail in the upstream documentation.

There are three different deployment configurations for Ceph Object Gateways, depending on requirements.

- Simple Ceph Object Gateway

  A simple Ceph Object Gateway configuration is used where the Ceph Object Storage service runs in a single data center and there is no requirement to define regions and zones.

- Multisite Ceph Object Gateway configuration is used where the Ceph Object Storage service is geographically distributed within a federated architecture and provides the facility to configure storage for different regions and to further distinguish separate zones per region. Data synchronization agents allow the service to maintain multiple copies of the data across a widely distributed environment, helping to provide better data resilience for failover, backup and disaster recovery. The feature facilitates the ability to write to non-master zones and maintain synchronization of data changes between different zones.

Note

Ceph Storage for Oracle Linux Release 1.0 required that you manually configure an external web server and the FastCGI module for use with the Ceph Object Gateway. In this release, the software includes an embedded version of the lightweight Civetweb Web Server that simplifies installation and configuration of the Ceph Object Gateway service.

1.6.1 Simple Ceph Object Gateway

The Ceph Object Gateway is a client of the Ceph Storage Cluster, but may be hosted on a node within the cluster if required. The Ceph Object Gateway has the following requirements:

- A running Ceph Storage Cluster
- A public facing network interface that allows traffic on the network port used to serve HTTP or HTTPS requests (7480 by default)
- A name for the Ceph Object Gateway instance
- A storage cluster user name with appropriate permissions in a keyring
- Pools to store its data
- A data directory for the gateway instance
- An instance entry in the Ceph Configuration file

The following sections describe installation and deployment steps to get you started using Ceph Object Gateway.

Deployment

To install the Ceph Object Gateway software on a node within your storage cluster, you can run the following command from the deployment node within your environment:

```
# ceph-deploy install --rgw ceph-node1
```

Substitute *ceph-node1* with the resolvable hostname of the node where you wish to install the software. Note that the target node must have the appropriate Yum channels configured, as described in Section 1.2, “Enabling Access to the Ceph Packages”.

To create a Ceph Object Gateway within the Ceph configuration, use the following command:

```
# ceph-deploy --overwrite-conf rgw create ceph-node1
```

Updating Firewall Requirements

If you are running a firewall service, make sure that the port where the Ceph Object Gateway is running is open. For example:

```
# firewall-cmd --zone=public --add-port 7480/tcp --permanent
# firewall-cmd --reload
```

Note that if you change the default port for the Ceph Object Gateway at a later stage, you may need to repeal this firewall rule and add a new rule for the new port number.

Creating Users and Keys

Before you are able to use the Ceph Object Gateway, users must be created to allow access to the different APIs exposed by the gateway.

To create a user for S3 access, run the following command on the gateway host:

```
# radosgw-admin user create --uid="testuser" --display-name="First User"
```

The command returns JSON formatted output describing the newly created user. This output includes a listing of the keys that are automatically generated when you create a new user. Take note of the `access_key` and `secret_key` for the user that you have created. You require these when connecting to the gateway from an S3 client application.

If you wish to provide access to the Swift API, a subuser can be created against the original user. To create a user for Swift access, run the following command on the gateway host:

```
# radosgw-admin subuser create --uid=testuser --subuser=testuser:swift
```

The output should display the details for the updated user, with the added subuser. Swift makes use of its own keys, so you need to create a specific key for this subuser before you are able to use it with a Swift client. To do this, run the following command:

```
# radosgw-admin key create --subuser=testuser:swift --key-type=swift --gen-secret
```

Once again, the details for the updated user are displayed. This time, the `swift_keys` key in the JSON output is updated to display a `user` and `secret_key` that can be used for Swift access validation.

At any point, if you need to see this user information again to obtain keys or to check other information, such as permissions, you can use the `radosgw-admin user info` command.

Testing Access

To test S3 access, you require the `python-boto` package and you must create a simple Python script that can be used to create a new bucket.

1. Install the `python-boto` package if it is not already installed:

   ```
   # yum install python-boto
   ```

2. Create a Python script that can be used to test S3 access. Using a text editor, create a file called `s3test.py` and insert the following code:

   ```
   #!/usr/bin/env python
   import boto
   import boto.s3.connection

   access_key = 'SZUP3NC5P7452N1HQT4B'
   secret_key = 'v0Ok4YK0MtcSZURk6vwCwRtQnB3vAW2G8TjrAlIj'
   conn = boto.connect_s3(
          aws_access_key_id = access_key,
          aws_secret_access_key = secret_key,
   ```

```
        host = 'ceph-node1.example.com', port = 7480,
        is_secure=False, calling_format = boto.s3.connection.OrdinaryCallingFormat(),
        )

bucket = conn.create_bucket('my-new-bucket')
for bucket in conn.get_all_buckets():
            print "{name} {created}".format(
                    name = bucket.name,
                    created = bucket.creation_date,
            )
```

Replace the `access_key` value, *`SZUP3NC5P7452N1HQT4B`*, with the access key for the *`testuser`* user that you created for S3 access. Replace the `secret_key` value, *`v0Ok4YK0MtcSZURk6vwCwRtQnB3vAW2G8TjrAlIj`*, with the secret key that was generated for the *`testuser`* user that you created for S3 access. Replace *`ceph-node1.example.com`* with the hostname or fully qualified domain name where the gateway host is located. Replace the port number *`7480`*, if you have configured an alternate port to the default.

3. Change permissions on the script so that you can run the script:

```
# chmod 776 ./s3test.py
```

4. Run the script:

```
# ./s3test.py
my-new-bucket 2016-09-19T09:25:17.056Z
```

The script should return the name of the new bucket and the date and timestamp for when it was created.

If you need to test Swift access, install the Swift command-line client and use the secret key that was generated for the subuser that you created for this purpose.

1. Install the `python-swiftclient` package and its dependencies:

```
# yum install python-swiftclient
```

2. Run the client from the command line, providing the appropriate credentials and connection information:

```
# swift -A http://ceph-node1.example.com:7480/auth/1.0 -U testuser:swift \
   -K '2DHaQknPsc5XsYEmHQ0mWCGLnoGnaCr4VUd62czm' list
my-new-bucket
```

Replace *`ceph-node1.example.com`* with the hostname or fully qualified domain name where the gateway host is located. Replace the port number *`7480`*, if you have configured an alternate port to the default. Replace *`testuser:swift`*, with the *`testuser`* user subuser that you created for Swift access. Replace *`2DHaQknPsc5XsYEmHQ0mWCGLnoGnaCr4VUd62czm`*, with the secret Swift key that was generated for the Swift subuser. Run `man swift` for more information about this command and the options available.

The command should list any existing buckets within Ceph, including the bucket created when you tested S3 access.

Changing Port Numbering

The port number that is used for the Ceph Object Gateway HTTP interface can be updated or changed in the configuration file on the admin node within the Storage Cluster. To change the port number used by the

Ceph Object Gateway, edit the `ceph.conf` file in your working directory on the deployment node of your cluster. Add the following lines to the end of the configuration file:

```
[client.rgw.ceph-node1]
rgw_frontends = "civetweb port=80"
```

Replace *`ceph-node1`* with the resolvable hostname that you used when you deployed the gateway. Replace *`80`* with the port number that you wish to use for the HTTP port.

To push the configuration change to the gateway node and to the other nodes in the cluster, run the following command:

```
# ceph-deploy --overwrite-conf config push ceph-node{1..4}.example.com
```

Note that in the above command, *`ceph-node{1..4}.example.com`* is equivalent to a space separated list that includes all of the nodes within the storage cluster, and specifically includes the gateway node.

On the gateway node, you should restart the Ceph Object Gateway for the settings to take effect:

```
# systemctl restart ceph-radosgw@*
```

Enabling SSL

To enable SSL on the Ceph Object Gateway service, you must install the OpenSSL packages on the gateway host, if they are not installed already:

```
# yum install -y openssl mod_ssl
```

Create an SSL certificate and key that can be used by the Ceph Object Gateway service. Instructions are provided in the workaround described for Section 1.8.5, “SSL SecurityWarning: Certificate has no `subjectAltName`”. Ideally, the certificate should be signed by a recognized Certificate Authority (CA).

Important

If you configure your Ceph Object Gateway to use a self-signed certificate, you may encounter SSL certificate verification or validation errors when you attempt to access the service in SSL mode, particularly when using the example python script provided in this document.

If you choose to use a self-signed certificate, you can copy the CA certificate to the client system's certificate bundle to avoid any errors. For example:

```
# cat custom.crt >> /etc/pki/tls/certs/ca-bundle.crt
```

Alternately, use the client program or script's environment to specify the path to additional trusted CA certificates in PEM format. The environment variables SSL_CERT_FILE and SSL_CERT_DIR can be used to specify additional trusted CA certificates. For example:

```
# SSL_CERT_FILE=/root/ceph/custom.pem python script.py
```

Note that Oracle does not recommend the use of self-signed certificates in production environments.

Update the `ceph.conf` file in your working directory on the deployment node of your cluster. If there is an existing entry for `[client.rgw.gateway]`, you may need to modify it to look similar to the following example. Alternatively add an entry that looks similar to the following:

```
[client.rgw.ceph-node1]
rgw_frontends = "civetweb port=443s ssl_certificate=/etc/pki/tls/ceph-node1.example.com.pem
```

Replace *ceph-node1* with the resolvable hostname that you used when you deployed the gateway. Replace *443* with the port number that you wish to use for the HTTPS port. Note that the port number must have the letter `s` affixed to indicate to the embedded Civetweb web server that HTTPS should be used on this port. Replace */etc/pki/tls/ceph-node1.example.com.pem* with the path to a PEM formatted file that contains both the certificate and key file.

To push the configuration change to the gateway node and to the other nodes in the cluster, run the following command:

```
# ceph-deploy --overwrite-conf config push ceph-node{1..4}.example.com
```

Note that in the above command, *ceph-node{1..4}.example.com* is equivalent to a space separated list that includes all of the nodes within the storage cluster, and specifically includes the gateway node.

On the gateway node, you should restart the Ceph Object Gateway for the settings to take effect:

```
# systemctl restart ceph-radosgw@*
```

If you are running firewall software on the gateway node, make sure that a rule exists to allow traffic on the port defined in your configuration file. For instance:

```
 # firewall-cmd --zone=public --add-port=443/tcp --permanent
```

1.6.2 Multisite Ceph Object Gateway

A multisite Ceph Object Gateway configuration can be deployed to achieve synchronization between different zones within a zonegroup. The multisite configuration replaces the federated Ceph Object Gateway configuration described in previous releases of Ceph.

Oracle has tested a multisite configuration consisting of a single zonegroup containing multiple zones distributed across separate storage clusters. No sync agent is configured to mirror data changes between gateways, which allows for a simpler active-active configuration. All metadata operations, such as the creation of new users, must be made via the master zone, however data operations, such as the creation of buckets and objects can be handled by any zone in the deployment.

The following configuration steps describe a basic multisite configuration consisting of two Ceph Storage Clusters in a single zone group containing three separate zones that actively sync data between them. The example setup is deployed on three servers with local storage available. It is assumed that the systems do not have an existing Ceph configuration and that they have access to the Ceph packages and their dependencies, as described in Section 1.2, “Enabling Access to the Ceph Packages”.

The following naming conventions are used in the example configuration:

- Realm: `gold`
- Master Zonegroup: `us`
- Master Zone: `us-east-1`
- Secondary Zone: `us-east-2`
- Secondary Zone: `us-west`

The zones `us-east-1` and `us-east-2` are part of the same Ceph Storage Cluster. The zone `us-west` is installed on a second Ceph Storage Cluster.

Setup of First Cluster

1. Install the ceph-deploy tool on one of the systems that shall be part of the first cluster:

```
# yum install ceph-deploy
```

   This system is referred to as the 'first cluster deployment node' through the rest of these instructions.

2. Create a clean working Ceph configuration directory for the storage cluster and change to this directory, for example:

```
# rm -rf /var/mydom_ceph
# mkdir /var/mydom_ceph
# cd /var/mydom_ceph
```

3. Clear the systems of any pre-existing Ceph configuration information or data:

```
# ceph deploy purge ceph-node1 ceph-node2
# ceph deploy purgedata ceph-node1 ceph-node2
```

   Replace *`ceph-node1`* and *`ceph-node2`* with the hostnames of the systems taking part in the cluster.

4. Deploy the Ceph cluster configuration:

```
# ceph-deploy new ceph-node1 ceph-node2
```

   Replace *`ceph-node1`* and *`ceph-node2`* with the hostnames of the systems taking part in the cluster.

5. Update the configuration template with required configuration variables:

```
# echo "osd pool default size = 2" >> ceph.conf
# echo "rbd default features = 3" >> ceph.conf
```

6. Install the Ceph cluster packages on the nodes:

```
# ceph-deploy install ceph-node1 ceph-node2
```

   Replace *`ceph-node1`* and *`ceph-node2`* with the hostnames of the systems taking part in the cluster.

7. Create a cluster monitor on one of the nodes:

```
# ceph-deploy mon create-initial
# ceph-deploy mon create ceph-node1
# ceph-deploy gatherkeys ceph-node1
```

   Replace *`ceph-node1`* with the hostname of the node that you wish to designate as a cluster monitor.

8. Prepare an available disk on each node to function as an Object Storage Device (OSD):

```
# ceph-deploy osd create --zap-disk --fs-type xfs ceph-node1:sdb
# ceph-deploy osd create --zap-disk --fs-type xfs ceph-node2:sdc
```

   Replace *`ceph-node1`* and *`ceph-node2`* with the hostnames of the systems taking part in the cluster. Replace *`xfs`* with your preferred filesystem type, either `xfs` or `btrfs`. Replace *`sdb`* and *`sdc`* with the appropriate device names for available disks on each host. Note that these disks or repartitioned and formatted, destroying any existing data on them.

9. Check the Ceph status to make sure that the cluster is healthy and that the OSDs are available:

```
# ceph status
```

Setup Ceph Object Gateway Instances on First Cluster

1. From the first cluster deployment node, install the Ceph Object Gateway software on each of the nodes in the cluster:

```
# ceph-deploy install --rgw ceph-node1 ceph-node2
# ceph-deploy rgw create ceph-node1 ceph-node2
```

   Replace *ceph-node1* and *ceph-node2* with the hostnames of the systems taking part in the cluster where you wish to install the Ceph Object Gateway software.

2. Edit the template configuration in `/var/mydom_ceph/ceph.conf` on the first cluster deployment node and add the following lines to the end of the configuration file:

```
osd pool default pg num = 100
osd pool default pgp num = 100
mon pg warn max per osd = 2100

[client.rgw.ceph-node1]
rgw_frontends = "civetweb port=80"

[client.rgw.ceph-node2]
rgw_frontends = "civetweb port=80"
```

   Replace *ceph-node1* and *ceph-node2* with the hostnames of the gateway systems.

3. Push the configuration to each of the nodes in the cluster:

```
# ceph-deploy --overwrite-conf config push ceph-node1 ceph-node2
```

4. On *each* of the nodes, restart the Ceph Object Gateway service and check its status to ensure that it is running correctly:

```
# systemctl restart ceph-radosgw@*
# systemctl status ceph-radosgw@*
```

Create Required Pools on First Cluster

Ceph Object Gateways require several pools to store gateway related data. Where gateways are configured as zones, it is typical to create pools particular to a zone using the naming convention: *zone.pool-name*. For this reason, it is best to manually create all of the required pools for each of the zones within the cluster.

On the first cluster deployment node, run the following commands to create all of the pools required for the zones that are hosted in this cluster:

```
# ceph osd pool create ceph-us-east-1.rgw.control 16 16
# ceph osd pool create ceph-us-east-1.rgw.data.root 16 16
# ceph osd pool create ceph-us-east-1.rgw.gc 16 16
# ceph osd pool create ceph-us-east-1.rgw.log 16 16
# ceph osd pool create ceph-us-east-1.rgw.intent-log 16 16
# ceph osd pool create ceph-us-east-1.rgw.usage 16 16
# ceph osd pool create ceph-us-east-1.rgw.users.keys 16 16
# ceph osd pool create ceph-us-east-1.rgw.users.email 16 16
# ceph osd pool create ceph-us-east-1.rgw.users.swift 16 16
# ceph osd pool create ceph-us-east-1.rgw.users.uid 16 16
# ceph osd pool create ceph-us-east-1.rgw.buckets.index 32 32
```

```
# ceph osd pool create ceph-us-east-1.rgw.buckets.data 32 32
# ceph osd pool create ceph-us-east-1.rgw.meta 16 16

# ceph osd pool create ceph-us-east-2.rgw.control 16 16
# ceph osd pool create ceph-us-east-2.rgw.data.root 16 16
# ceph osd pool create ceph-us-east-2.rgw.gc 16 16
# ceph osd pool create ceph-us-east-2.rgw.log 16 16
# ceph osd pool create ceph-us-east-2.rgw.intent-log 16 16
# ceph osd pool create ceph-us-east-2.rgw.usage 16 16
# ceph osd pool create ceph-us-east-2.rgw.users.keys 16 16
# ceph osd pool create ceph-us-east-2.rgw.users.email 16 16
# ceph osd pool create ceph-us-east-2.rgw.users.swift 16 16
# ceph osd pool create ceph-us-east-2.rgw.users.uid 16 16
# ceph osd pool create ceph-us-east-2.rgw.buckets.index 32 32
# ceph osd pool create ceph-us-east-2.rgw.buckets.data 32 32
# ceph osd pool create ceph-us-east-2.rgw.meta 16 16
```

Create System Keys

While configuring zones, each gateway instance requires a system user with credentials set up to allow for S3-like access. This allows each gateway instance to pull the configuration remotely using the access and secret keys. To make sure that the same keys are configured on each gateway instance, it is best to define these keys beforehand and to set them manually when the zones and users are created.

It is good practice to set these as reusable environment variables while you are setting up your configuration and to randomize the keys as much as possible:

```
# SYSTEM_ACCESS_KEY=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 20 | head -n 1)
# SYSTEM_SECRET_KEY=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 40 | head -n 1)
```

You can check that these keys are set and contain good content:

```
# echo SYSTEM_ACCESS_KEY=$SYSTEM_ACCESS_KEY
# echo SYSTEM_SECRET_KEY=$SYSTEM_SECRET_KEY
```

Keep a record of the output from these commands. You need to export the same environment variables when you set up the second cluster and the secondary zone located here.

Configure the Realm and Master Zone

The multisite configuration is built around a single realm, named `gold`, with a single zonegroup called `us`. Within this zonegroup is a master zone named `ceph-us-east-1`. The following steps describe what must be done to create and configure these components:

1. Create the realm and make it the default:

   ```
   # radosgw-admin realm create --rgw-realm=gold --default
   ```

2. Delete the default zonegroup which is created as part of the simple installation of the Ceph Object Gateway software.

   ```
   # radosgw-admin zonegroup delete --rgw-zonegroup=default
   ```

3. Create a new master zonegroup. The master zonegroup is in control of the zonegroup map and propagates changes across the system. This zonegroup should be set as the default zonegroup to allow you to run commands for it in future without having to explicitly identify is using the `--rgw-zonegroup` switch.

   ```
   # radosgw-admin zonegroup create --rgw-zonegroup=us \
    --endpoints=http://ceph-node1.example.com:80 --master --default
   ```

4. Create the master zone and make it the default zone. Note that for metadata operations, such as user creation, you must use this zone. You can also add the zone to the zonegroup when you create it, and specify the access and secret key that should be used for this zone:

```
# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=ceph-us-east-1 \
 --endpoints=http://ceph-node1.example.com:80 --access-key=$SYSTEM_ACCESS_KEY \
 --secret=$SYSTEM_SECRET_KEY --default --master
```

5. Create a system user that can be used to access the zone pools. The keys for this user must match the keys used by each each of the zones that are being configured:

```
# radosgw-admin user create --uid=zone.user --display-name="ZoneUser" \
  --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY --system
```

6. The realm period holds the entire configuration structure for the current state of the realm. When realm information, such as configuration for the zonegroups and zones, is modified, the changes must be updated for the period. This is achieved by committing the changes:

```
# radosgw-admin period update --commit
```

Configure Secondary Zone

The following commands can be executed on the first cluster deployment node, but are used to update the zonegroup and realm configuration to add the secondary zone hosted on the other node within the cluster.

1. Create the secondary zone, making sure that you specify the same access and secret key as used for the master zone:

```
# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=ceph-us-east-2 \
    --endpoints=http://ceph-node2.example.com:80 \
    --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY
```

2. Update the realm period with the new configuration information:

```
# radosgw-admin period update --commit
```

Update Ceph Configuration and Restart the Gateways

Edit the template configuration in the working directory on the first cluster deployment node to map the zone names to each gateway configuration. This is done by adding a line for the `rgw_zone` variable to the gateway configuration entry for each node:

```
[client.rgw.ceph-node1]
rgw_frontends = "civetweb port=80"
rgw_zone=ceph-us-east-1

[client.rgw.ceph-node2]
rgw_frontends = "civetweb port=80"
rgw_zone=ceph-us-east-2
```

When you have updated the template configuration, push the changes to each of the nodes in the cluster:

```
# ceph-deploy --overwrite-conf config push ceph-node1 ceph-node2
```

On *each* of the nodes, restart the Ceph Object Gateway service and check its status to ensure that it is running correctly:

```
# systemctl restart ceph-radosgw@*
# systemctl status ceph-radosgw@*
```

Setup of Second Cluster

Install and deploy a second cluster in much the same way as you did the first. In this example, the second cluster consists of a single node, although you may add more nodes if you require. This node ultimately hosts the gateway for the `ceph-us-west` zone.

The following commands, recap on the steps to deploy the cluster and to configure an OSD that can be used for storage. These commands must be issued on a new server, *`ceph-node3`*, outside of the first cluster:

```
# mkdir -p /var/mydom_ceph; cd /var/mydom_ceph
# yum install ceph-deploy

# ceph-deploy new ceph-node3

# echo "osd pool default size = 2" >> ceph.conf
# echo "rbd default features = 3" >> ceph.conf

# ceph-deploy install ceph-node3
# ceph-deploy mon create-initial
# ceph-deploy mon create ceph-node3
# ceph-deploy gatherkeys ceph-node3

# ceph-deploy osd create --zap-disk --fs-type xfs ceph-node3:sdb
```

Setup Ceph Object Gateway Instance on Second Cluster

1. Install the Ceph Object Gateway software on the newly deployed node in the cluster:

   ```
   # ceph-deploy install --rgw ceph-node3
   # ceph-deploy rgw create ceph-node3
   ```

   Replace *`ceph-node3`* with the hostname of the node where you wish to install the Ceph Object Gateway software.

2. Edit the template configuration in `/var/mydom_ceph/ceph.conf` on the second cluster deployment node and add the following lines to the end of the configuration file:

   ```
   osd pool default pg num = 100
   osd pool default pgp num = 100
   mon pg warn max per osd = 2100

   [client.rgw.ceph-node3]
   rgw_frontends = "civetweb port=80"
   ```

   Replace *`ceph-node3`* with the hostname of the gateway system.

3. Push the configuration to each of the nodes in the cluster:

   ```
   # ceph-deploy --overwrite-conf config push ceph-node3
   ```

4. Restart the Ceph Object Gateway service on the gateway node and check its status to ensure that it is running correctly:

   ```
   # systemctl restart ceph-radosgw@*
   # systemctl status ceph-radosgw@*
   ```

Create Required Pools on Second Cluster

Create the required pools for the Ceph Object Gateway on the second cluster by running the following commands:

```
# ceph osd pool create ceph-us-west.rgw.control 16 16
# ceph osd pool create ceph-us-west.rgw.data.root 16 16
# ceph osd pool create ceph-us-west.rgw.gc 16 16
# ceph osd pool create ceph-us-west.rgw.log 16 16
# ceph osd pool create ceph-us-west.rgw.intent-log 16 16
# ceph osd pool create ceph-us-west.rgw.usage 16 16
# ceph osd pool create ceph-us-west.rgw.users.keys 16 16
# ceph osd pool create ceph-us-west.rgw.users.email 16 16
# ceph osd pool create ceph-us-west.rgw.users.swift 16 16
# ceph osd pool create ceph-us-west.rgw.users.uid 16 16
# ceph osd pool create ceph-us-west.rgw.buckets.index 32 32
# ceph osd pool create ceph-us-west.rgw.buckets.data 32 32
# ceph osd pool create ceph-us-west.rgw.meta 16 16
```

Update Realm Configuration

1. Export the same `SYSTEM_ACCESS_KEY` and `SYSTEM_SECRET_KEY` environment variables that you set up on the first cluster. For example:

```
# SYSTEM_ACCESS_KEY=OJywnXPrAA4uSCgv1UUs
# SYSTEM_SECRET_KEY=dIpf1FRPwUYcXfswYx6qjC0eSuHEeHy0I2f9vHFf
```

2. Using these keys, pull the realm configuration directly from the first cluster, via the node running the master zone, by issuing the following command:

```
# radosgw-admin realm pull --url=http://ceph-node1.example.com:80 \
  --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY
```

3. Pull the period state directly from the first cluster, via the node running the master zone:

```
# radosgw-admin period pull --url=http://ceph-node1.example.com:80 \
  --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY
```

4. Set the default realm for the gateway instance to `gold`:

```
# radosgw-admin realm default --rgw-realm=gold
```

5. Set the default zonegroup to `us`:

```
# radosgw-admin zonegroup default --rgw-zonegroup=us
```

Configure Secondary Zone

1. Create the new secondary zone, *`ceph-us-west`*, and add it to the *`us`* zonegroup. Make sure that when you create the zone you use the same access and secret keys as were used on the original configuration on the first cluster:

```
# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=ceph-us-west \
 --endpoints=http://ceph-node3.example.com:80 \
 --default --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY
```

2. Commit the zonegroup changes to update the period state:

```
# radosgw-admin period update --commit --rgw-zone=ceph-us-west
```

3. Edit the template configuration in the working directory at `/var/mydom_ceph/ceph.conf` to map the zone names to the gateway configuration. This is done by adding a line for the `rgw_zone` variable to the gateway configuration entry:

```
[client.rgw.ceph-node3]
rgw_frontends = "civetweb port=80"
```

```
rgw_zone=ceph-us-west

[client.rgw.ceph-node2]
rgw_frontends = "civetweb port=80"
rgw_zone=ceph-us-east-2
```

4. When you have updated the template configuration, push the changes to each of the nodes in the cluster:

```
# ceph-deploy --overwrite-conf config push ceph-node3
```

Restart the Ceph Object Gateway service and check its status to ensure that it is running correctly:

```
# systemctl restart ceph-radosgw@*
# systemctl status ceph-radosgw@*
```

Test Zone Synchronization

At this point all zones should be running and synchronizing.

To test synchronization, you can create a bucket in any of the zones and then list the buckets on any alternative zone. You should discover that the newly created bucket is visible within any of the zones.

This test can be performed using a simple Python script. Copy the example script, included below, into a file called `~/s3zone_test.py` on any host that is able to access each of the nodes where the zones are running:

```
#!/usr/bin/env python

import boto
import boto.s3.connection
from optparse import OptionParser

parser = OptionParser()
parser.add_option("--access_key", dest="access_key", default="OJywnXPrAA4uSCgv1UUs")
parser.add_option("--secret_key", dest="secret_key", default="dIpf1FRPwUYcXfswYx6qjC0eSuHEeHy0I2f9vHFf")
parser.add_option("-H","--host", dest="host", default="ceph-node1.example.com")
parser.add_option("-s","--secure",dest="is_secure", action="store_true", default=False)
parser.add_option("-c","--create", dest="bucket")

(options,args) = parser.parse_args()

conn = boto.connect_s3(
 aws_access_key_id = options.access_key,
 aws_secret_access_key = options.secret_key,
 host = options.host,
 is_secure=options.is_secure,
 calling_format = boto.s3.connection.OrdinaryCallingFormat(),
)
if options.bucket:
    bucket = conn.create_bucket(options.bucket)

for bucket in conn.get_all_buckets():
        print "{name}\t{created}".format(
                name = bucket.name,
                created = bucket.creation_date,
)
```

Substitute the default values for the `access_key`, `secret_key` and `host` in the script to better suit your own environment.

To test the script, make sure that it is executable by modifying the permissions on the script:

```
# chmod 775 ~/s3zone_test.py
```

Check that you have all the appropriate libraries installed to properly run the script:

```
# yum install python-boto
```

You can use the options in the script to set different variables such as which zone host you wish to run the script against, and to determine whether or not to create a new bucket. Create a new bucket in the first zone by running the following command:

```
# ~/s3zone_test.py --host ceph-node1 -c my-bucket-east-1
my-bucket-east-1        2016-09-21T09:16:14.894Z
```

Now check the other two nodes to see that the new bucket is synchronized:

```
# ~/s3zone_test.py --host ceph-node2
my-bucket-east-1        2016-09-21T09:16:16.932Z
# ~/s3zone_test.py --host ceph-node3
my-bucket-east-1        2016-09-21T09:16:15.145Z
```

Note that the timestamps are different due to the time that it took to synchronize the data. You may also test creating a bucket in the zone located on the second cluster:

```
# ~/s3zone_test.py --host ceph-node3 -c my-bucket-west-1
my-bucket-east-1        2016-09-21T09:16:15.145Z
my-bucket-west-1        2016-09-21T09:22:15.456Z
```

Check that this bucket is synchronized into the other zones:

```
# ~/s3zone_test.py --host ceph-node1
my-bucket-east-1        2016-09-21T09:16:14.894Z
my-bucket-west-1        2016-09-21T09:22:15.428Z
# ~/s3zone_test.py --host ceph-node2
my-bucket-east-1        2016-09-21T09:16:16.932Z
my-bucket-west-1        2016-09-21T09:22:17.488Z
```

Configure SSL

When configuring SSL for a multisite Ceph Object Gateway deployment, it is critical that each zone is capable of validating and verifying the SSL certificates. This means that if you choose to use self-signed certificates, each zone must have a copy of all of the certificates already within its recognized CA bundle. Alternatively, make sure that you use certificates signed by a recognized Certificate Authority.

Note that you may wish to use the instructions that are provided in the workaround described for Section 1.8.5, “SSL SecurityWarning: Certificate has no `subjectAltName`” when creating your certificates to avoid the warning message mentioned.

The steps provided in this example show how to create self-signed certificates for each gateway node, how to share the generated certificates between the nodes to ensure that they can be validated and how to change the existing multisite configuration to enable SSL.

1. Create certificates for each gateway node in the deployment.

   On each gateway node, run the following commands:

   ```
   # cd /etc/ceph
   # openssl genrsa -out ca.key 2048
   # openssl req -new -key ca.key -out ca.csr -subj "/C=US/ST=California/L=RedWoodShore/O=Oracle/OU=Linux/CN=
   ```

```
# openssl x509 -req -days 365 -in ca.csr -signkey ca.key -out ca.crt
# cp -f ca.crt /etc/pki/tls/certs/`hostname`.crt
# cp -f ca.key /etc/pki/tls/private/`hostname`.key
# cp -f ca.csr /etc/pki/tls/private/`hostname`.csr
# cp ca.crt /etc/pki/tls/`hostname`.pem
# cat ca.key >> /etc/pki/tls/`hostname`.pem
```

You may replace the values for the certificate subject line with values that are more appropriate to your organization, but ensure that the CommonName (CN) of the certificate resolves to the hostname that you use for the endpoint URLs in your zone configuration.

2. Copy the certificates to from each gateway node to the collection of recognized CAs in `/etc/pki/tls/certs/ca-bundle.crt` on each node.

   For example, on *ceph-node1*:

   ```
   # cat /etc/ceph/ca.crt >> /etc/pki/tls/certs/ca-bundle.crt
   # cat /etc/ceph/ca.crt | ssh root@ceph-node2 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   # cat /etc/ceph/ca.crt | ssh root@ceph-node3 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   ```

   On *ceph-node2*:

   ```
   # cat /etc/ceph/ca.crt >> /etc/pki/tls/certs/ca-bundle.crt
   # cat /etc/ceph/ca.crt | ssh root@ceph-node1 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   # cat /etc/ceph/ca.crt | ssh root@ceph-node3 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   ```

   On *ceph-node3*:

   ```
   # cat /etc/ceph/ca.crt >> /etc/pki/tls/certs/ca-bundle.crt
   # cat /etc/ceph/ca.crt | ssh root@ceph-node1 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   # cat /etc/ceph/ca.crt | ssh root@ceph-node2 "cat >> /etc/pki/tls/certs/ca-bundle.crt"
   ```

3. If you are running a firewall service on any of the nodes in your environment, make sure that traffic is permitted on port 443. For example:

   ```
   # firewall-cmd --zone=public --add-port=443/tcp --permanent
   # systemctl restart firewalld.service
   ```

4. Modify the existing zonegroup information to use HTTPS to access any zone endpoints.

   To do this run the following commands:

   ```
   # radosgw-admin zonegroup get|sed -r 's/http(.*):80/https\1:443/g' >/tmp/zonegroup.json
   # radosgw-admin zonegroup set --infile /tmp/zonegroup.json
   # radosgw-admin period update --commit
   ```

5. Redeploy the Ceph Object Gateway on each node in your environment, to reset any previous configuration and to ensure that the nodes are deployed using the full hostname matching the CN used for the certificates.

   Run the following command from the first cluster deployment node:

   ```
   # ceph-deploy --overwrite-conf rgw create ceph-node1.example.com ceph-node2.example.com
   ```

   Substitute *ceph-node1.example.com* and *ceph-node2.example.com* with the full hostnames of the nodes that are running the gateway software, so that these match the CN used on their certificates.

   Run the following command from the second cluster deployement node:

   ```
   # ceph-deploy --overwrite-conf rgw create ceph-node3.example.com
   ```

Substitute *ceph-node3.example.com* with the full hostname of the node that is running the gateway software, so that it matches the CN used on the certificate that you generated for this node.

At this point, all of the gateway services should have restarted running on the default port 7480.

6. On the deployment node on each cluster, edit the template configuration to change the port number and to identify the location of the SSL certificate PEM file for each gateway.

   For example, on *ceph-node1*:

```
# cd /var/mydom_ceph
```

   Edit `ceph.conf` and modify the gateway configuration entries. Make sure that the full hostname is used in the entry label and that you modify the port and add an entry pointing to the SSL certificate path:

```
...
osd pool default pg num = 100
osd pool default pgp num = 100
mon pg warn max per osd = 2100

[client.rgw.ceph-node1.example.com]
rgw_frontends = "civetweb port=443s ssl_certificate=/etc/pki/tls/ceph-node1.example.com.pem"
rgw_zone=ceph-us-east-1

[client.rgw.ceph-node2.example.com]
rgw_frontends = "civetweb port=443s ssl_certificate=/etc/pki/tls/ceph-node2.example.com.pem"
rgw_zone=ceph-us-east-2
```

   Push the modified configuration to each gateway node:

```
# ceph-deploy --overwrite-conf config push ceph-node1 ceph-node2
```

   On *each* of the nodes, restart the Ceph Object Gateway service and check its status to ensure that it is running correctly:

```
# systemctl restart ceph-radosgw@*
# systemctl status ceph-radosgw@*
```

   Repeat this step on the second cluster deployment node, to update the configuration for *ceph-node3.example.com*.

7. At this point, each gateway entry should be configured to use SSL. You can test that the zones are continuing to synchronize correctly and are using SSL, by using the test script at `~/s3zone_test.py`, remembering to use the `-s` switch to enable SSL.

   For example:

```
# ~/s3zone_test.py -s --host ceph-node2.example.com -c my-bucket-east-2
my-bucket-east-1        2016-09-21T09:16:16.932Z
my-bucket-east-2        2016-09-21T14:09:51.287Z
my-bucket-west-1        2016-09-21T09:22:17.488Z
# ~/s3zone_test.py -s --host ceph-node3.example.com
my-bucket-east-1        2016-09-21T09:16:15.145Z
my-bucket-east-2        2016-09-21T14:09:58.783Z
my-bucket-west-1        2016-09-21T09:22:15.456Z
```

   If you attempt to use the script without the `-s` switch set, the script attempts to connect without SSL on port 80 and fails to connect, ultimately terminating with a socket error:

```
socket.error: [Errno 111] Connection refused
```

1.7 Ceph FS

Important

The Ceph FS feature is available as a technology preview. Support for this feature is limited and it should not be used in production environments. If you intend to experiment with this feature, refer to the upstream documentation for more detailed information: http://docs.ceph.com/docs/master/cephfs/.

1. Deploy a Ceph Metadata Server (MDS).

   At least one metadata server must be active within your environment to use Ceph FS. If you do not have a dedicated server available for this purpose, you may install the MDS service on an existing Monitor node within your Storage Cluster, as the service does not have significant resource requirements. To deploy Ceph MDS in your environment, execute the following command from your deployment node:

   ```
   # ceph-deploy mds create ceph-node3
   ```

2. Deploy the Ceph Client.

   The Ceph Client must be deployed on the system where you intend to mount the Ceph FS and this system must have the appropriate network access and authentication keyring to access the Storage Cluster. See Section 1.5.1, “Installing the Ceph Client” for more information on deploying and setting up the Ceph Client.

3. Create storage pools and a new Ceph Filesystem.

   A Ceph Filesystem requires at least two storage pools to function. The first pool is used to store actual data, while the second is used to store metadata. Although the Ceph command line includes commands for creating and removing Ceph filesystems, only one filesystem can exist at the same time.

   To create the storage pools, run the following commands from the Ceph Client node:

   ```
   # ceph osd pool create cephfs_data 1
   # ceph osd pool create cephfs_metadata 2
   ```

   To create the new Ceph Filesystem, run the following command from the Ceph Client node:

   ```
   # ceph fs new cephfs cephfs_metadata cephfs_data
   ```

4. Check the status of the Ceph MDS.

   After a filesystem is created, the Ceph MDS enters into an *active* state. You are only able to mount the filesystem once the MDS is active. To check its status:

   ```
   # ceph mds stat
   e5: 1/1/1 up {0=ceph-node3=up:active}
   ```

5. Store the secret key used for admin authentication into a file that can be used for mounting the Ceph FS.

   The Storage Cluster admin key is stored in `/etc/ceph/ceph.client.admin.keyring` on each node in the cluster and also on the Ceph Client node. When mounting a Ceph FS the key is required

to authenticate the mount request. To prevent this key from being visible in the process list, it is best practice to copy it into a file that is secured with the appropriate permissions to keep this information safe.

For example:

```
# echo $(sed -n 's/.*key *= *\([^ ]*.*\)/\1/p' < /etc/ceph/ceph.client.admin.keyring) > /etc/ceph/admin.sec
# chmod 600 /etc/ceph/admin.secret

# cat /etc/ceph/ceph.client.admin.keyring
[client.admin]
        key = AQDIvtZXzBriJBAA+3HmoYkUmPFnKljqhxlYGw==
# cat /etc/ceph/admin.secret
AQDIvtZXzBriJBAA+3HmoYkUmPFnKljqhxlYGw==
```

6. To mount the filesystem, you can either use the Ceph kernel module or you may use the Ceph FUSE (Filesystem in User Space) tools.

   Mount using kernel module

   a. Make the directory mount point, if required:

   ```
   # mkdir -p /mnt/cephfs
   ```

   b. Mount the file system:

   ```
    # mount -t ceph ceph-node2:6789:/ /mnt/cephfs -o name=admin,secretfile=/etc/ceph/admin.secret
   ```

   Replace *ceph-node2* with the hostname or IP address of a Ceph monitor within your Storage Cluster. Multiple monitor addresses can be specified, separated by commas, although only one active monitor is needed to successfully mount the filesystem. If you do not know what monitors are available, you can run `ceph mon stat` to get a listing of available monitors, their IP addresses and port numbers.

   Mounting a Ceph Filesystem automatically loads the `ceph` and `libceph` kernel modules on the Ceph Client node.

   c. To unmount the filesystem:

   ```
   # umount /mnt/cephfs
   ```

   Mount using Ceph FUSE

   a. If you have not installed the ceph-fuse package already, install it on the client system:

   ```
   # yum install ceph-fuse
   ```

   b. Ceph FUSE can be used from any system, as long as it has access to the Ceph Storage Cluster configuration information and it has a copy of the Ceph Client admin keyring. If you have already configured this host as a Ceph Client, this information is already available and you may skip this step.

   - On the client host, create the appropriate configuration directory:

     ```
     # mkdir -p /etc/ceph
     ```

   - Copy the configuration and admin keyring to this directory from one of the Ceph Storage Cluster monitor hosts:

```
# scp /etc/ceph/ceph.conf root@client_host:/etc/ceph
# scp /etc/ceph/ceph.client.admin.keyring root@client_host:/etc/ceph
```

- Ensure that the Ceph configuration file and the keyring have the appropriate permissions set on the client system:

```
# chmod -R 644 /etc/ceph/
```

c. Create a mount point for the file system, if required:

```
# mkdir -p /mnt/cephfs
```

d. To mount the Ceph Filesystem as FUSE, use the `ceph-fuse` command. For example:

```
# ceph-fuse -c /etc/ceph/ceph.conf /mnt/cephfs
```

e. To unmount a Ceph Filesystem mounted as FUSE, do the following:

```
# fusermount -u /mnt/cephfs
```

1.8 Known Issues

The following sections describe known issues in this release.

1.8.1 The `ceph-deploy` tool is not compatible with Ceph "Firefly" release packages

The `ceph-deploy` tool, provided in this release of Ceph, is not compatible with the previous Ceph release packages. Using this tool in conjunction with an existing Ceph deployment that is based on the Ceph "Firefly" release can result in errors and in unexpected behavior. This is because the components within this release are significantly different from previous releases in terms of their packaging, configuration and architecture. Do not attempt to use this tool to install or configure components from a previous release.

(Bug 24579333)

1.8.2 Syslog errors about missing file during some OSD operations

During some disk operations on an OSD, such as when creating a new disk or when zapping a disk, an error appears in the syslog indicating that a file could not be found. For example:

```
systemd-udevd: error: /dev/sde2: No such file or directory
 systemd-udevd: inotify_add_watch(7, /dev/sde2, 10) failed: No such file or directory
```

This error appears due to the asynchronous behavior of partition creation and the triggered udev event. The udev event is triggered while the disk partition is still in the process of updating but has not yet completed. The error is harmless and can be ignored, as the OSD operation completes correctly.

(Bug 24716988)

1.8.3 RBD kernel module fails to map an image to a block device

Many of the features supported by Ceph for image formats on a Rados Block Device are not yet supported by the kernel module included with UEK R4 update 2. This can cause problems when mapping an image to a block device resulting in the following error:

```
# rbd map vol1 --pool datastore
```

```
rbd: sysfs write failed
RBD image feature set mismatch. You can disable features unsupported by the
kernel with "rbd feature disable".
In some cases useful info is found in syslog - try "dmesg | tail" or so.
rbd: map failed: (6) No such device or address
```

This requires that when the image is created the appropriate features are enabled manually or that you change the default feature set in the Ceph configuration.

Workaround: To resolve this issue, edit `/etc/ceph/ceph.conf` to include the line:

```
rbd default features = 3
```

Alternatively enable the layering feature when creating the image, by using the `--image-feature` switch. For example:

```
# rbd create vol1 --size 250 --pool datastore --image-feature layering
```

(Bug 23562267)

1.8.4 Ceph Object Gateway does not support HTTP and HTTPS concurrently

Although upstream documentation suggests that it is possible to configure the Ceph Object Gateway to support both HTTP and HTTPS at the same time, by specifying two ports concatenated with the `+` symbol and by appending the `s` character to the port number where SSL/TLS should be used, this functionality does not work properly and only the first port specified is used.

Only configure for one protocol at a time, until this issue is resolved.

(Bug 24422558)

1.8.5 SSL SecurityWarning: Certificate has no `subjectAltName`

When you configure a Ceph Object Gateway instance and enable SSL you must create an SSL certificate. If the certificate does not have the v3 extension enabled and the subjectAltName set within the certificate, a warning message is displayed when a client such as the Swift client attempts to access the gateway:

```
/usr/lib/python2.7/site-packages/urllib3/connection.py:251: SecurityWarning:
Certificate has no `subjectAltName`, falling back to check for a `commonName`
for now. This feature is being removed by major browsers and deprecated by
RFC 2818. (See https://github.com/shazow/urllib3/issues/497 for details.)
```

If a `subjectAltName` extension of type `dNSName` is present, this is used as the identity. Otherwise, the `Common Name` field in the `Subject` field of the certificate is used. Although the use of the Common Name is existing practice, it is deprecated and Certification Authorities are encouraged to use the dNSName instead.

To prevent the warning from appearing at all, do the following:

1. In the working directory where you are generating the key and certificate, create a copy of the template OpenSSL configuration file:

   ```
   # cp /etc/pki/tls/openssl.cnf ./
   ```

2. Modify the configuration file template at `./openssl.cnf` and make the following changes:

   - In the section `[ req ]` make sure that the following line is uncommented and not preceded with a # character:

```
req_extensions = v3_req # The extensions to add to a certificate request
```

- In the section `[ v3_req ]`, add the following line to the end of the parameters in this section:

```
subjectAltName = @alt_names
```

- Add a section to the end of the configuration file:

```
[ alt_names ]
DNS.1 = hostname.example.com
```

Replace *`hostname.example.com`* with the fully qualified domain name for the host that you are creating the certificate for.

3. Generate your certificate key, as normal:

```
# openssl genrsa -out hostname.example.com.key 2048
```

4. Use the certificate key and the new `openssl.cnf` file to create a Certificate Signing Request (CSR):

```
# openssl req -new -key hostname.example.com.key \
-out hostname.example.com.csr -extensions v3_req -config openssl.cnf
```

5. You may either use the generated CSR to obtain a signed certificate from a recognized Certificate Authority (CA). Or, for testing purposes, you may use this to generate a self-signed certificate as follows:

   - Create a new configuration file, `v3.cnf`, that can host the information for the v3 requirements. Edit it to contain the following lines:

```
[v3_req]
subjectAltName = @alt_names
[alt_names]
DNS.1 = hostname.example.com
```

   - Run the following OpenSSL command to generate a self-signed certificate using the CSR and your local key:

```
# openssl x509 -req -days 365 -in hostname.example.com.csr -signkey hostname.example.com.key \
> -out hostname.example.com.crt -extensions v3_req -extfile v3.cnf
```

6. Copy the key, CSR and certificate to the usable location on the host:

```
# cp -f hostname.example.com.crt /etc/pki/tls/certs/
# cp -f hostname.example.com.csr /etc/pki/tls/private/
# cp -f hostname.example.com.key /etc/pki/tls/private/
```

7. Create a single PEM file containing both the key and certificate, that can be used by the Ceph Object Gateway when it is started:

```
# cp hostname.example.com.crt hostname.example.com.pem
# cat hostname.example.com.key >> hostname.example.com.pem
# cp hostname.example.com.pem /etc/pki/tls/
```

(Bug 24424028)

1.8.6 SELinux policy errors during installation of `ceph-selinux` package

The following errors may appear in the syslog for any system where the ceph-selinux package is installed:

```
kernel: SELinux: Permission audit_read in class capability2 not defined in policy.
kernel: SELinux: Class binder not defined in policy.
kernel: SELinux: the above unknown classes and permissions will be allowed
```

The ceph-selinux policy was designed for an older kernel and is unaware of newer object classes and access vector permissions and are, therefore, ignored and allowed by default. This is expected behavior and does not affect or alter the policy in any way. The messages can be ignored.

(Bug 24289420)

1.8.7 High CPU load on multisite Ceph Object Gateway nodes

Ceph Object Gateway multisite configurations make significant use of `curl` functions such as `curl_multi_wait()`. A deadlock issue existed in earlier versions of the `curl` library, but was fixed in version 7.31 and later.

Oracle has backported the patch to fix this issue in `curl` version 7.29.0-25.0.2. This fix is available in the latest errata release of Oracle Linux 7.

If this problem occurs, make sure that you have the latest version of the `curl` and `libcurl` packages installed from the `ol7_latest` and `ol7_optional_latest` repositories.

(Bug 24355753)

1.8.8 Federated Ceph Object Gateway configuration does not work

The Multisite Ceph Object Gateway is designed to replace the original federated configuration and provides many fixes and improvements to the original design. While the upstream documentation still provides instruction on the federated configuration, many of the commands no longer work as documented and this configuration remains untested by Oracle.

Oracle does not support the federated configuration.

Ceph Terminology

Block Device
: A Ceph component that provides access to Ceph storage as a thinly provisioned block device. When an application writes to a Block Device, Ceph implements data redundancy and enhances I/O performance by replicating and striping data across the Storage Cluster.

  Also known as a *RADOS Block Device* or *RBD*.

Ceph OSD
: A Ceph component that provides access to an OSD.

  Also known as a *Ceph OSD Daemon*.

Client
: A host that can access the data stored in a Storage Cluster. A Ceph Client need not be a member node of a Storage Cluster.

Monitor (MON)
: A Ceph component used for tracking active and failed nodes in a Storage Cluster.

Node
: A system that is a member of a Storage Cluster.

Object Gateway
: A Ceph component that provides a RESTful gateway that can use the Amazon S3 and OpenStack Swift compatible APIs to present OSD data to Ceph Clients, OpenStack, and Swift clients. An Object Gateway is configured on a node of a Storage Cluster.

  Also known as a *RADOS Gateway* or *RGW*.

Object Storage Device (OSD)
: Storage on a physical device or logical unit (LUN). Typically, data on an OSD is configured as a btrfs file system to take advantage of its snapshot features. However, other file systems such as XFS can also be used.

Storage Cluster
: A Ceph component that stores MON and OSD data across cluster nodes.

  Also known as a *Ceph Object Store*, *RADOS Cluster*, or *Reliable Autonomic Distributed Object Store*.