Red Hat Openstack Platform

Integration with Ceph

Configuring Cinder, Glance and Libvirt to use Ceph Block Storage

Keith D Schincke <kschinck@redhat.com>

Version 1.0-pre

XXXX XX, 2014

100 East Davie Street
Raleigh NC 27601 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Comments and Feedback

In the spirit of open source, we invite anyone to provide feedback and comments on any reference architectures. Although we review our papers internally, sometimes issues or typographical errors are encountered. Feedback allows us to not only improve the quality of the papers we produce, but allows the reader to provide their thoughts on potential improvements and topic expansion to the papers.

Feedback on the papers can be provided by emailing refarch-feedback@redhat.com. Please refer to the title within the email.

Staying In Touch

Join us on some of the popular social media sites where we keep our audience informed on new reference architectures as well as offer related information on things we find interesting.

Like us on Facebook:

https://www.facebook.com/rhrefarch

Follow us on Twitter:

https://twitter.com/RedHatRefArch

Plus us on Google+:

https://plus.google.com/u/0/b/114152126783830728030/

Table of Contents

1 Executive Summary

This document provides a detailed set of instructions on how to integrate Red Hat Openstack Platform (OSP) with Ceph. The OSP controllers are configured to be highly available (HA). The OSP storage back end is configured to use Ceph storage via the RBD client library. Ceph pools will be created for OSP images, volumes and backups. All node access will be controlled via the cephx authentication protocol.

2 Component Overview

2.1 What is Openstack?

OpenStack is an open-source project for building a private or public Infrastructure-as-a-Service (IaaS) cloud running on standard hardware. The typical analogy is that OpenStack gives you public, cloud-like capabilities in your datacenter.

OpenStack is a cloud operating system that controls all the infrastructure — compute, storage, and networking—resources throughout a datacenter, all managed through a central dashboard called Horizon.

2.2 What is Red Hat OpenStack?

Red Hat Enterprise Linux OpenStack Platform delivers an integrated foundation to create, deploy, and scale a secure and reliable public or private OpenStack cloud. It delivers a cloud platform built on Red Hat Enterprise Linux, optimized for and integrated with Red Hat's OpenStack technologies, providing the agility to scale and quickly meet customer demands without compromising on availability, security, or performance.

2.3 Nova

Nova is the project name for OpenStack Compute, a cloud computing fabric controller; the main part of an IaaS system. Individuals and organizations can use Nova to host and manage their own cloud computing systems.

2.4 Glance

The Glance project provides services for discovering, registering, and retrieving virtual machine images. Glance has a RESTful API that allows querying of VM image metadata as well as retrieval of the actual image.

2.5 Cinder

The OpenStack Block Storage service provides compute instances with persistent block storage. Block storage is appropriate for performance sensitive scenarios, such as databases or frequently-accessed file systems. Persistent block storage can survive instance termination. It can also be moved between instances like any external storage device. This service can be backed by a variety of enterprise storage platforms or simple NFS servers. Cinder's features include:

- Persistent block storage devices for compute instances
- Self-service volume creation, attachment, and deletion
- A unified interface for numerous storage platforms
- Volume snapshots

2.6 Ceph

Ceph is an open-source, massively scalable, software-defined storage system which uniquely delivers object, block and file system storage in one unified system. 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. Ceph delivers extraordinary scalability–thousands of clients accessing exabytes of data. A Ceph Node leverages commodity hardware and intelligent daemons, and a Ceph Storage Cluster accommodates large numbers of nodes, which communicate with each other to replicate and redistribute data.

2.7 Ceph Storage Cluster

The Ceph Storage Cluster is the foundation for all Ceph deployments. Based upon RADOS (Reliable Autonomic Distributed Object Store), Ceph Storage Clusters consist of two types of daemons: a Ceph OSD Daemon (OSD) stores data as objects on a storage node, and a Ceph Monitor maintains a master copy of the cluster map. A Ceph Storage Cluster may contain thousands of storage nodes. A minimal system will have at least one Ceph Monitor and at least two Ceph OSD Daemons for data replication.

Ceph stores data objects within two logical groups: pools and placement groups (PGs).

- **Pools:** Pools are logical groups for storing objects. Pools manage the number of placement groups, the number of replicas, and the ruleset for the pool. To store data objects in a pool, you must have an authenticated user with permissions for the pool. Ceph can snapshot pools too!

  Each pool has a number of placement groups. CRUSH (Controlled Replication Under Scalable Hashing) maps PGs to OSDs dynamically. When a Ceph Client stores objects, CRUSH will map each object to a placement group. Mapping objects to placement groups creates a layer of indirection between the Ceph OSD and the Ceph Client.
- **Placement Groups:** Ceph Storage Clusters may store countless objects (e.g., tens of millions of objects), but must be able to grow (or shrink) and rebalance dynamically. If the Ceph Client "knew" which Ceph OSD Daemon had which object via a look-up table, which would create a single point of failure. If the Ceph Client stored which Ceph OSD Daemon had which object, it would create a tight coupling between the Ceph Client and the Ceph OSD Daemon, precluding fault tolerance. Instead, the CRUSH algorithm maps each object to a placement group, and then maps each placement group to one or more Ceph OSD Daemons. This layer of indirection allows Ceph to rebalance efficiently when new Ceph OSD Daemons and the underlying OSD devices come online (or crash and go offline).

2.8 Ceph Block Device

A Ceph block device provides a block-based storage interface to the Ceph Storage Cluster. The ubiquity of block device interfaces makes a virtual block device an ideal candidate to interact with a mass data storage system like Ceph. Ceph block devices are thin-provisioned, resizable, provide copy-on-write cloning and store block device data as objects striped over multiple OSDs in a Ceph cluster. Ceph block devices leverage RADOS capabilities, such as snapshotting, replication, and consistency. OpenStack deployments that use Ceph as a back end for volumes and images use libvirt and QEMU/KVM as an interface to Ceph's RADOS Block Devices (RBD) library, librbd.

2.9 Inktank Ceph Enterprise (ICE)

Inktank Ceph Enterprise™ combines the most stable version of open-source Ceph® for object and block storage with a Ceph management platform, enhanced integration tools, and a suite of support services. The flagship offering of Inktank, acquired recently by Red Hat, Inktank Ceph Enterprise provides everything enterprises need to confidently run production Ceph clusters at scale, thereby radically improving the economics and management of storage and providing a foundation for managing the exponential growth in enterprise data. Inktank Ceph Enterprise customers benefit from hardened software with long-term support, lower TCO, predictable costs, and expert services from Ceph's core developers.

3 Implementation

3.1 Ceph Pre-Installation Configuration Requirements

A number of configuration settings are suggested across the Ceph administration nodes. The next sections provide a recommended set of configuration options. These options should be tailored to meet local system configuration and operational requirements.

3.1.1 Time synchronization

Clock synchronization is very important for a functioning Ceph cluster. It is recommended that all hosts use the same time reference. This configuration can be performed during the kickstart process with the following configuration option and should be updated for local configuration and administrative rights.

```
timezone America/New_York --isUtc –ntpservers=XXX.XXX.XXX.XXX
```

3.1.2 Non-root Administrative User

A non-root administrative user can be used to perform the Ceph configuration commands. This user name should be available across all of the hosts. The account can exist in a central account database or individually.

The user can be created during the kickstart process with the following configuration options and should be updated for local configuration and administrative rights.

```
user --groups=admin --name=ceph-user --password=XXXXXXXX
```

3.1.3 Sudo Access

Sudo access needs to be granted if the non-root administrative is used to execute Ceph configuration commands. Below is an example command to allow members of the admin group to execute commands via sudo. The configuration settings should be updated to satisfy local administrative and configuration requirements.

```
# echo “Defaults:%admin !requiretty” >> /etc/sudoers
# echo “%admin   ALL=(ALL)  NOPASSWD:  ALL” >> /etc/sudoers
```

3.1.4 Limiting SSH Access

Secure Shell (SSH) can be updated to restrict access to a restricted list of users or groups. The below commands will permit members of the admin group to ssh into the server. These commands should be tailored to satisfy local administrative requirements and ensure proper access to the host.

Note: Running these commands unmodified may prevent the root user from logging in remotely. Provisions should be added to allow other account access as needed.

```
# echo “AllowGroups admin” >> /etc/ssh/sshd_config
# systemctl restart sshd
```

3.1.5 SSH Key Authentication

An SSH authentication key pair needs to be created for the Ceph administration users on the ICE administration host. The below commands will generate a key pair and copy them to each of the Ceph servers. The key size and type should be adjusted as needed to satisfy local administrative requirements.

Note: In this example, the $CEPH_HOST_LIST variable is replaced with a list of all of the Ceph hostnames.

```
[ceph-user@ice-admin ~]$ ssh-keygen
[ceph-user@ice-admin ~]$ for HOST in $CEPH_HOST_LIST; do ssh-copy-id $HOST;
done
```

3.1.6 Firewall Configuration Changes

Network port and protocol access requirements are provided in this document. These access rules need to be modified for local administrative and configuration requirements. Any required remote administration, application, and monitoring access requirements should also be maintained. The below command will add and save a firewall rule to the start of the INPUT chain in order to allow SSH connections.

```
# iptables -I INPUT 1 -p tcp --dport 22 -j ACCEPT
# service iptables save
```

3.2 Ceph Installation

3.2.1 Ceph Requirements

- Access to ICE 1.2.2 release and updated ceph-deploy package
- Ceph nodes installed with RHEL7 (document developed with 7.0)
- SSH access allowed between the ICE Administration host and all Ceph hosts

- SSH key access allowed from ICE Administration host to all Ceph Storage Nodes
- All host clocks are in alignment
- All Ceph hosts are subscribed to the server yum repo groups (rhel-x86_64-server-7)
- All host packages are up to date
- Selinux is in permissive mode on all Ceph hosts. BZ 1127910 and BZ 1144559 are opened to track these issues

3.2.2 Ceph Configuration Goal

Multiple networks as defined in Table 3.2.2.1: Network Configuration are used in this example configuration. IP are assigned consistently on hosts requiring access to multiple networks. An /etc/hosts file is used for hostname resolution on the storage network. This solution does not scale, and a larger name resolution system is recommended for larger or production deployments.

Name	Range	Description
Provisioning	10.19.0.0/16	Server installation, updates and configuration Ceph-deploy connectivity calamari monitoring
Storage	172.31.0.0/16	Ceph Client to Ceph communication Ceph monitor to OSD communication
Storage Cluster	172.30.0.0/16	Ceph OSD to OSD communication

Table 3.2.2.1: Network Configuration

Table 3.2.2.2: Ceph Hosts details the name of each host, its purpose, IP address, and required network connectivity.

Host	Purpose	IP	Network Connectivity
Ospfm	Openstack Foreman	10.19.141.60	Provisioning
ospctl1	Ceph Mon, OSP HA Controller	10.19.139.31, 172.31.139.31	Provisioning, Storage
ospctl2	Ceph Mon, OSP HA Controller	10.19.139.32, 172.31.139.32	Provisioning, Storage
ospctl3	Ceph Mon, OSP HA controller	10.19.139.33, 172.31.139.33	Provisioning, Storage
ospcomp1	OSP Compute	10.19.139.34, 172.31.139.34	Provisioning, Storage
ospcomp2	OSP Compute	10.19.139.35, 172.31.139.35	Provisioning, Storage
ice-admin	ICE Administration	10.19.141.51	Provisioning
ceph-osd1	Ceph OSD	10.19.141.55, 172.31.141.55, 172.30.141.55	Provisioning, Storage, Storage Cluster
ceph-osd2	Ceph OSD	10.19.141.56, 172.31.141.56, 172.30.141.56	Provisioning, Storage, Storage Cluster
ceph-osd3	Ceph OSD	10.19.141.57, 172.31.141.57, 172.30.141.57	Provisioning, Storage, Storage Cluster

Table 3.2.2.2: Ceph Hosts

In this example configuration, multiple OSDs are configured per Ceph storage host. A journal disk is shared between multiple OSD partitions in a ratio of 1 to 5. The OSDs sharing a journal device should be placed within the same CRUSH failure domain. Table 3.2.2.3: OSD Configuration details the OSD host data and journal drive configuration.

Note: Partitions are automatically created on the Journal Device by the ceph-deploy command. There should be unallocated available space to support the creation of a journal partition per OSD.

OSD Host	Journal Device	Data Paths
ceph-osd1	/dev/sdl	/dev/sdb, /dev/sdc, /dev/sdd, /dev/sde, /dev/sdf
	/dev/sdm	/dev/sdg, /dev/sdh, /dev/sdi, /dev/sdj, /dev/sdk
ceph-osd2	/dev/sdl	/dev/sdb, /dev/sdc, /dev/sdd, /dev/sde, /dev/sdf
	/dev/sdm	/dev/sdg, /dev/sdh, /dev/sdi, /dev/sdj, /dev/sdk
ceph-osd3	/dev/sdl	/dev/sdb, /dev/sdc, /dev/sdd, /dev/sde, /dev/sdf
	/dev/sdm	/dev/sdg, /dev/sdh, /dev/sdi, /dev/sdj, /dev/sdk

Table 3.2.2.3: OSD Configuration

3.2.3 Deploy ICE And The Calamari Server

1. Enable web access to the ICE Admin server for downloading Ceph supplied packages.

   Note: tcp:443 support must be added if https is selected for the calamari web server port below.

```
[ceph-user@ice-admin ~]$ sudo iptables -I INPUT 1 -p tcp --dport 22 -j
ACCEPT
[ceph-user@ice-admin ~]$ sudo service iptables save
```

2. Enable access to the salt master ports used by the calamari service to administer the Ceph servers.

```
[ceph-user@ice-admin ~]$ sudo iptables -I INPUT 1 -p tcp --dport 4505 -j
ACCEPT
[ceph-user@ice-admin ~]$ sudo iptables -I INPUT 1 -p tcp --dport 4506 -j
ACCEPT
[ceph-user@ice-admin ~]$ sudo service iptables save
```

3. Download and extract the ICE distribution package. The ICE distribution package may be copied from a local location or downloaded from the vendor website at https://download.inktank.com/enterprise/.

   Note: Login credintals are required to download the packaged distribution file.

```
[ceph-user@ice-admin ~]$ mkdir ~/ice-1.2
[ceph-user@ice-admin ~]$ cd ~/ice-1.2
[ceph-user@ice-admin ice-1.2]$ #Copy or download file
[ceoh-user@ice-admin ice-1.2]$ tar -zxvf ICE-1.2.2-rhel7.tar.gz
```

4. Run the ICE deploy script.

```
[ceph-user@ice-admin ice-1.2]$ sudo python ice_setup.py
```

   Note: The cpehdeploy.conf file is created in the same directory as the ice_setup.py file. A ~/.cephdeploy.conf file is created with the same contents and is used by the ceph-deploy command.

5. Type 'yes' to continue the installation.
6. Press [ENTER] to accept the FQDN hostname or alter as needed. This hostname needs to be accessible to the Ceph nodes on the provisioning network.
7. Select http or https as need. http is used in this example configuration.
8. Create a ceph-deploy working directory. All ceph-deploy commands should be executed from this directory.

```
[ceph-user@ice-admin ~]$ mkdir ~/cluster && cd ~/cluster
```

9. Run calamari-ctl initialize.

```
[ceph-user@ice-admin cluster]$ sudo calamari-ctl initialize
```

10. Enter the calamari login name. “root” is the default but it can be changed to satisfy local administrative requirements.
11. Enter the email address to be associated with the admin account.
12. Enter the administrator password.
13. Re-enter the administrator password. This username and password is used to log into the calamari GUI for cluster monitoring and administration.

3.2.4 Ceph Monitor Configuration Process

1. The /etc/hosts file on the ICE Administration host and the generated ceph.conf file must be need to be manually updated, since ICE Administration host does not have access to the storage network and the ceph-deploy command assumes access to this network. With this change, the ICE Administration host will use the storage network names of hosts to access via the provisioning network.

```
[ceph-user@ice-admin ~]# tail -3 /etc/hosts
10.19.139.71 ospctl1-storage
10.19.139.72 ospctl2-storage
10.19.139.73 ospctl3-storage
```

2. Create the ceph cluster with an initial monitor list

```
[ceph-user@ice-admin cluster]$ ceph-deploy new ospctl1-storage \
 ospctl2-storage ospctl3-storage
```

3. Update or add any parameters in the ceph.conf file as needed to satisfy local operational requirements. In this example configuration, the “mon_host”, “public network” and “cluster network” parameters are added or modified to match the local configuration conditions. The “osd pool default size” parameter is set to 2 in order to allow the initial cluster to be configured with two storage nodes. Appendix C: Example Configuration Files includes the ceph.conf file used for this example deployment.

```
[ceph-user@ice-admin cluster]# diff ceph.conf.ORIG ceph.conf
4c4
< mon_host = 10.19.139.71,10.19.139.72,10.19.139.73
---
> mon_host = 172.31.139.71,172.31.139.72,172.31.139.73
9c9,11
<
---
> public network = 172.31.0.0/16
> cluster network = 172.30.0.0/16
> osd pool default size = 2
```

4. Deploy the Ceph software to the initial monitor node.

```
[ceph-user@ice-admin cluster]$ ceph-deploy install  ospctl1 ospctl2 \
ospctl3
```

5. On initial start up, the Ceph initialization script attempts to configure the keys for and bind to the IP address for the hostname of the host. The hostname of each Ceph monitor host must be temporarily changed to the storage network hostname.

   Note: The hostname can be reset once the initial quorum is reached below.

```
[ceph-user@ice-admin cluster]$ ssh ospctl1 sudo hostname ospctl1-storage
[ceph-user@ice-admin cluster]$ ssh ospctl2 sudo hostname ospctl2-storage
[ceph-user@ice-admin cluster]$ ssh ospctl3 sudo hostname ospctl3-storage
```

6. On each monitor, enable access to the Ceph monitor process.

```
[ceph-user@ospctl1 ~]$ sudo iptables -I INPUT 1 -p tcp --dport 6789 -j
ACCEPT
[ceph-user@ospctl1 ~]$ sudo service iptables save
[ceph-user@ospctl2 ~]$ sudo iptables -I INPUT 1 -p tcp --dport 6789 -j
ACCEPT
[ceph-user@ospctl2 ~]$ sudo service iptables save
[ceph-user@ospctl3 ~]$ sudo iptables -I INPUT 1 -p tcp --dport 6789 -j
ACCEPT
[ceph-user@ospctl3 ~]$ sudo service iptables save
```

7. Create the initial monitors.

```
[ceph-user@ice-admin cluster]$ ceph-deploy mon create-initial
```

8. On a monitor host, verify Ceph operation from one of the monitors. This page includes information on how to monitor the health of a Ceph cluster. Any issues should be resolved prior to proceeding farther in this document.

```
[root@ospctl1 ~]$ sudo ceph health
HEALTH_ERR 192 pgs stuck inactive; 192 pgs stuck unclean; no osds;
[root@ospctl1 ~]$ ceph mon stat
e2: 3 mons at
{ospctl1-storage=172.31.139.71:6789/0,ospctl2-storage=172.31.139.72:6789/
0,ospctl3-storage=172.31.139.73:6789/0}, election epoch 8, quorum 0,1,2
ospctl1-storage,ospctl2-storage,ospctl3-storage
```

   NOTE: A HEALTH_ERR status is expected in this case as there are no OSDs configured and the placement groups have not been replicated. There should be three monitors listed and the cluster should have a quorum.

9. Once a monitor quorum is reached, the hostnames of the Ceph monitors can be reset to normal.

```
[ceph-user@ice-admin cluster]$ ssh ospctl1 sudo hostname \
`cat /etc/hostname`
[ceph-user@ice-admin cluster]$ ssh ospctl2 sudo hostname \
`cat /etc/hostname`
[ceph-user@ice-admin cluster]$ ssh ospctl3 sudo hostname \
`cat /etc/hostname`
```

3.2.5 Ceph OSD Configuration Process

Prepare the OSDs on each host according to Table 3.2.2.4: OSD Configuration. The commands for ceph-osd1 are shown below. The configuration commands for ceph-osd2, ceph-osd3, or additional storage servers are similar.

1. Deploy the Ceph software to the OSD hosts.

```
[ceph-user@ice-admin cluster]$ ceph-deploy install ceph-osd1
```

2. The Ceph OSD opens three network ports per OSD starting at TCP:6800. For example, if 5 OSDs are configured on a server then 15 TCP ports are needed. If 10 OSDs are configured then 30 ports are needed. Additional ports are suggested to allow daemon restart.

```
[ceph-user@ceph-osd1 ~]$ sudo iptables -I INPUT 1 -p tcp -m multiport
--dports 6800-6840 -j ACCEPT
[ceph-user@ceph-osd1 ~]$ sudo service iptables save
```

3. Llist all disks attached to the OSD server

```
[ceph-user@ice-admin cluster]$ ceph-deploy osd disk list ceph-osd1
```

4. Clear any existing partitions on a device which will be used as an OSD or journal

   Note: Do not run this command against the device hosting the operating system or other needed inofrmation.

```
[ceph-user@ice-admin cluster]$ ceph-deploy osd disk zap \
ceph-osd1:/dev/sdb
```

5. Partition and format the data disks and allocate space for the journal.

   Note: The OSD data paths are mounted when the Ceph OSD process starts. The default mount path is /var/lib/ceph/osd/{cluster}-{osd}. The path can be overridden via the “osd data” configuration parameter.

```
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdb:/dev/sdl
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdc:/dev/sdl
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdd:/dev/sdl
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sde:/dev/sdl
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdf:/dev/sdl
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdg:/dev/sdm
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdh:/dev/sdm
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
```

```
ceph-osd1:/dev/sdi:/dev/sdm
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdj:/dev/sdm
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd1:/dev/sdk:/dev/sdm
```

6. Verify OSD status.

   Note: the health status should be HEALTH_OK and all OSDs should be up. Troubleshoot any issues prior to continuing.

```
[ceph-user@ospctl1 ~]$ sudo ceph health
HEALTH_OK
[ceph-user@ospctl1 ~]$ sudo ceph osd tree
# id  weight      type name  up/down    reweight
-1    27.3  root default
-2    9.1         host ceph-sd1
0     0.91              osd.0 up    1
1     0.91              osd.1 up    1
2     0.91              osd.2 up    1
.... Output truncated
```

3.2.6 Connecting Ceph hosts to the calamari server

1. Use ceph-deploy to connect hosts to the calamari server.

```
[ceph-user@ice-admin cluster]$ ceph-deploy calamari connect ospctl1 \
 ospctl2 ospctl3 ceph-osd1 ceph-osd2
```

2. Using the login credentials from the 3.2.3 Deploy ICE And The Calamari Server, log into the calamari web GUI.

```
[ceph-user@ice-admin cluster]$ firefox http://ceph-osd1
```

3. Add all hosts pending approval.
4. At this point, ICE, Ceph and calamari are installed and in a default configuration. Additional configuration items must be performed as needed to satisfy local operational needs.

3.3 Add Ceph OSD hosts

3.3.1 Requirements

- All configuration requirements of Section 3.3.1 met
- Configuration items from Section 3.2 completed

3.3.2 Ceph Installation

1. From the ICE Administration host, install Ceph packages onto the new host.

```
[ceph-user@ice-admin cluster]$ ceph-deploy install ceph-osd3
```

2. Prepare and activate the OSD disks available on host as required by system requirements.

```
[ceph-user@ice-admin cluster]$ ceph-deploy disk list ceph-osd3
```

```
[ceph-user@ice-admin cluster]$ ceph-deploy osd create \
ceph-osd3:/dev/sdb:/dev/sdl
```

3. Verify OSD status.

```
[ceph-user@ospctl1 ~]$ sudo ceph health
HEALTH_OK
[ceph-user@ospctl1 ~]$ sudo ceph osd tree
```

4. Placement group rebalancing can be monitored via the calamari GUI via the 'ceph status' command from a host with administrative rights.

   Note: The number of placement groups may need to be adjusted as the number of OSDs increases.

5. Connect the new OSD to the calamari administration server.

```
[ceph-user@ice-admin cluster]$ ceph-deploy calamari connect ceph-osd3
```

3.3.3 Granting Ceph Administration Rights To A Non-monitor Host

By default the ceph-deploy will grant Ceph administrative rights to the monitor hosts. These rights can be granted to other hosts to facilitate centralized management.

1. Run the following commands to grant administrative rights to one of the OSD hosts

```
[ceph-user@ice-admin cluster]$ ceph-deploy admin ceph-osd1
```

Note: The admin host needs access to the Storage Network. The ICE Admin host in this document does not have access to this network.

2. Verify Admin access

```
[ceph-user@ceph-osd1 ~]$ sudo ceph status
```

3.3.4 Modifying OSD Placement Groups

The number of placement groups (pg) and placement groups for placement (pgp) may require adjusting as the number of OSDs increase in the cluster, or the number of replicas change for a pool. The number of placement groups depends on the number of OSDs, the number of replicas and is the result of the calculation (number OSD * 100) / number of replicas rounded up to the nearest power of 2. In this example, there are 20 OSDs and 2 replicas with 1024 being the nearest power of 2. See this link for information on how to calculate placement groups.

Note: If the number of placement groups needs to be increased by a large amount, then the change should be made in multiple steps.

For example, for a Ceph cluster with 10 OSDs and two replicas for each pool. The number of placement groups will be calculated as:

(10 * 100) / 2 = 500 rounded to 512.

If the number of OSDs in increased to 30 then the number of placement groups will be calculated as:

(30 * 100) / 2 = 1500 rounded to 2048

1. Increase the number of placement groups for the data pool, run the following command from any Ceph host with administrative rights.

```
[ceph-user@ospctl1 ~]$ sudo ceph osd pool set data pg_num 2048
```

2. Increase the number of placement groups for placement.

```
[ceph-user@ospctl1 ~]$ sudo ceph osd pool set data pgp_num 2048
```

3.4 OSP Integration with Ceph

3.4.1 Integration Requirements

- Foreman admin server installed and functioning
- Deployed and functional Ceph ICE 1.2.X installation
- Base RHEL7 installed on OSP controller and compute nodes with puppet running
- ceph-deploy commands are run from the ICE configuration directory
- SSH access allowed between the ICE Administration host and all OSP hosts
- SSH key access allowed from ICE Administration host to all OSP hosts
- Selinux must be in permissive mode on all OSP nodes
- Hosts are configured according to Table 3.2.2-1: Ceph Hosts

3.4.2 Integration Configuration Goal

Three Ceph pools will be created: images, volumes and (optional) backups.

OSP will be installed and configured to store data within the created pools via Foreman/puppet.

3.4.3 Ceph Pool Creation And Keyring Configuration

1. From any host with Ceph administrative rights, create Ceph pools for volumes, images and backups. The number of placement groups depends on the number of OSDs and the number of replicas and is the result of the calculation (number OSD * 100) / number of replicas rounded up to the nearest power of 2. In this example, there are 20 OSDs and 2 replicas with 1024 being the nearest power of 2. See this link for information on how to calculate placement groups.

```
[ceph-user@ospctl1 ~]$ sudo ceph osd pool create images 1024
[ceph-user@ospctl1 ~]$ sudo ceph osd pool create volumes 1024
[ceph-user@ospctl1 ~]$ sudo ceph osd pool create backups 1024
```

2. List available pools.

```
[ceph-user@ospctl1 ~]$ sudo ceph osd lspools
```

3. Create, populate and import the keyring for images pool.

   Note: The first command line must be one continuous line.

```
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.images mon
```

```
'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx
pool=images'
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.images > \
/etc/ceph/ceph.client.images.keyring
```

4. Create, populate and import the keyring for volumes pool.

   Note: The first command line must be one continuous line.

```
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.volumes mon
'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx
pool=volumes, allow rx pool=images'
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.volumes > \
/etc/ceph/ceph.client.volumes.keyring
```

5. If OSP backups are to be configured, create, populate and import the keyring for backups pool.

   Note: The first command line must be one continuous line.

```
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.backups mon
'allow r' osd 'allow class-read object_prefix rbd_children, allow rwx
pool=backups'
[ceph-user@ospctl1 ~]$ sudo ceph auth get-or-create client.backups > \
/etc/ceph/ceph.client.backups.keyring
```

6. List ceph secret keys and capabilities. “client.images”, “client.volumes” and “client.backups” should be listed. The keys will be different but the capabilities should be as below.

```
[ceph-user@ospctl1 ceph]$ sudo ceph auth list
.... truncated output ....
client.backups
        key: A...==
        caps: [mon] allow r
        caps: [osd] allow class-read object_prefix rbd_children, allow
rwx pool=backups
client.images
        key: A...==
        caps: [mon] allow r
        caps: [osd] allow class-read object_prefix rbd_children, allow
rwx pool=images
client.volumes
        key: A...==
        caps: [mon] allow r
        caps: [osd]allow class-read object_prefix rbd_children, allow rwx
pool=volumes, allow rx pool=images
```

7. Add the paths for the new client keyrings to the */etc/ceph/ceph.conf* file.

```
[ceph-user@ospctl1 ~]$ tail -6 ceph.conf
[client.images]
      keyring = /etc/ceph/ceph.client.images.keyring
[client.volumes]
      keyring = /etc/ceph/ceph.client.volumes.keyring
[client.backups]
      keyring = /etc/ceph/ceph.client.backups.keyring
```

8. Pull the new configuration file to the ICE Administration host.

```
[ceph-user@ice-admin cluster]$ ceph-deploy –overwrite-conf config \
```

```
    pull ospctl1
```

9. Copy the client keyrings to the ICE Administration host.

```
[ceph-user@ice-admin cluster]$ ssh ospctl1 “sudo cat
/etc/ceph/ceph.client.backups.keyring” >  ceph.client.backups.keyring
[ceph-user@ice-admin cluster]$ ssh ospctl1 “sudo cat
/etc/ceph/ceph.client.images.keyring” >  ceph.client.images.keyring
[ceph-user@ice-admin cluster]$ ssh ospctl1 “sudo cat
/etc/ceph/ceph.client.volumes.keyring” >  ceph.client.volumes.keyring
```

10. Deploy new config file to all configured OSP controller and Ceph nodes.

```
[ceph-user@ice-admin cluster]$ ceph-deploy --overwrite-conf config \
push ospctl2 ospctl3 ceph-osd1 ceph-osd2 ceph-osd3
```

11. Copy the */etc/ceph/ceph.client.images.keyring*, */etc/ceph/ceph.client.volumes.keyring* and */etc/ceph/ceph.client.backups.keyring* to each monitor’s */etc/ceph* directory.

```
[ceph-user@ice-admin cluster]$ scp ceph.client.backups.keyring \
 ceph.client.images.keyring ceph.client.volumes.keyring \
 ospctl2:
[ceph-user@ice-admin cluster]$ ssh ospctl2 “sudo cp \
ceph.client.images.keyring /etc/ceph “
[ceph-user@ice-admin cluster]$ ssh ospctl2 “sudo cp \
ceph.client.volumes.keyring /etc/ceph “
[ceph-user@ice-admin cluster]$ scp ceph.client.backups.keyring \
 ceph.client.images.keyring ceph.client.volumes.keyring \
 ospctl3:
[ceph-user@ice-admin cluster]$ ssh ospctl3 “sudo cp \
ceph.client.images.keyring /etc/ceph “
[ceph-user@ice-admin cluster]$ ssh ospctl3 “sudo cp \
ceph.client.volumes.keyring /etc/ceph “
```

3.4.4 Foreman Configuration – HA All In One Controller

The following values need to be configured in the HA All In One Controller host group. Additional settings will need to be configured for the local environment.

The UUID value is replaced with the output of uuidgen or a similar UUID generation program.

Class	Variable	Value
quickstack::pacemaker::cinder	backend_rbd	true
	rbd_secret_uuid	UUID
quickstack::pacemaker::glance	backend	rbd
	pcmk_fs_manage	false

Table 3.4.4.1: HA All In One Controller Settings

3.4.5 Foreman Configuration – Compute (Nova Network)

The following values need to be configured in the Compute (Nova Network) host group. Additional settings will need to be configured for the local environment. The UUID value is replaced with the value generated in 3.4.4 Foreman Configuration – HA All In One Controller

Class	Variable	Value
quickstack::nova_network::compute	cinder_backend_rbd	true
	rbd_secret_uuid	UUID

Table 3.4.5.1: Compute (Nova Network) Settings

3.4.6 libvirt Configuration

1. On the Ceph administration node, extract the volumes client key.

```
[ceph-user@ice-admin cluster]$ cat ceph.client.volumes.keyring | grep \
key | awk ‘{print $3}’| tee client.volumes.key
```

2. On the Ceph administration node, create a *secret.xml* file containing the following contents.

   Note: GENERATED_UUID is replaced with the UUID created in 3.4.4 Foreman Configuration – HA All In One Controller.

```
<secret ephemeral='no' private='no' >
<uuid>GENERATED_UUID</uuid>
<usage type='ceph'>
<name>client.volumes secret</name>
</usage>
</secret>
```

3. Copy the *secret.xml* file and *client.volumes.key* file to each compute node.

```
[ceph-user@ice-admin cluster]$ scp secret.xml client.volumes.key \
 ospcomp1:
[ceph-user@ice-admin cluster]$ scp secret.xml client.volumes.key \
 ospcomp2:
```

4. On each compute node, define the secret for libvirt.

```
[ceph-user@ospcomp1 ~]$ sudo virsh secret-define --file secret.xml
```

5. On each compute node, use the UUID to add the contents of the secret key. Replace UUID command argument with the UUID from the file.

```
[ceph-user@ospcomp1 ~]$ sudo virsh secret-set-value --secret UUID \
--base64 `cat client.volumes.key`
```

6. On each compute node, print the virsh secrets and verify the new secret is added.

```
[ceph-user@ospcomp1 ~]$ sudo virsh secret-list
```

7. On each compute node, the *client.volumes.key* and *secret.xml* files can be removed at this point.

```
[ceph-user@ospcomp1 ~]$ rm client.volumes.key secret.xml
```

3.4.7 Deploy Ceph Configuration Files To Compute Hosts

Since Foreman does not manage the Ceph configuration files, the */etc/ceph/ceph.conf* will need to be deployed to each compute host.

1. Deploy */etc/ceph/ceph.conf* to each compute host.

```
[ceph-user@ice-admin cluster]$ ceph-deploy config push ospcomp1
[ceph-user@ice-admin cluster]$ ceph-deploy config push ospcomp2
```

2. Copy the *ceph.client.images.keyring* and *client.volumes.keyring*. Copy the *ceph.client.backups.keyring* file if the backup service is configured.

```
[ceph-user@ice-admin cluster]$ scp ceph.client.images.keyring \
ceph.client.volumes.keyring ospcomp1
[ceph-user@ice-admin cluster]$ scp ceph.client.images.keyring \
ceph.client.volumes.keyring ospcomp2
```

3. Place the keyring files in the */etc/ceph* directory.

```
[ceph-user@ice-admin cluster]$ ssh ospcomp1 sudo cp \
ceph.cient.images.keyring ceph.client.volumes.keyring /etc/ceph
[ceph-user@ice-admin cluster]$ ssh ospcomp2 sudo cp \
ceph.cient.images.keyring ceph.client.volumes.keyring /etc/ceph
```

3.4.8 Configure Cinder For Backups (Optional)

These configuration steps are only needed if the cinder backup service is configured.

1. On each controller node, set the below values in */etc/cinder/cinder.conf.*

```
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_conf /etc/ceph/ceph.conf
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_user backups
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_chunk_size 134217728
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_pool backups
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_stripe_unit 0
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_ceph_stripe_count 0
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT restore_discard_excess_bytes true
[ceph-user@ospctl1 ~]$ sudo openstack-config --set \
/etc/cinder/cinder.conf DEFAULT backup_driver cinder.backup.drivers.ceph
```

2. Restart cinder backup.

```
[ceph-user@ospctl1 ~]$ sudo pcs resource disable openstack-cinder-backup
[ceph-user@ospctl1 ~]$ sudo pcs resource enable openstack-cinder-backup
```

3.4.9 Test OSP Integration

Additional steps may be required depending on local configuration needs.

1. Create an OS image. (Test the images pool).

```
[root@ospctl1 ~]# glance image-create ...
```

2. List available images.

```
[root@ospctl1 ~]# glance image-list
```

3. Boot virtual image.

```
[root@ospctl1 ~]# nova boot ...
```

4. List running images.

```
[root@ospctl1 ~]# nova list
```

5. Create a new volume (Test the volumes pool).

```
[root@ospctl1 ~]# cinder create ...
```

6. List images.

```
[root@ospctl1 ~]# nova volume-list
```

7. Attach to running virtual image.

```
[root@ospctl1 ~]#  nova volume-attach ...
```

8. Log into image and verify access by formatting.

```
[root@inst1 ~]# fdisk /dev/vdb
```

9. Make a backup of the volume (Test the backups pool).

```
[root@ospctl1 ~]# cinder backup-create ....
```

10. List backups.

```
[root@ospctl1 ~]# cinder backup-list
```

Appendix A: Revision History

Revision 1.0	XXXXXXX XXX XX, 2014	Keith Schincke
Initial Release		

Appendix B: Command History

ICE Install

```
$ mkdir ice-1.2
$ cd ice-1.2/
$ #copy or download distribution file
$ tar -zxvf ICE-1.2.2-rhel7.tar.gz
$ sudo python ice_setup.py
$ sudo calamari-ctl initialize
```

Ceph Monitor Configuration

```
$ ceph-deploy new ospctl1-storage ospctl2-storage ospctl3-storage
$ vi ceph.conf
$ ceph-deploy install ospctl1-storage ospctl2-storage ospctl3-storage
$ ceph-deploy mon create-initial
```

Configuring the OSDs

```
$ ceph-deploy install ceph-osd1
$ ceph-deploy disk list ceph-osd1
$ ceph-deploy osd create ceph-osd1:/mnt/sdb:/dev/sdl
$ ceph-deploy osd create ceph-osd1:/mnt/sdc:/dev/sdl
```

Connect to calamari

```
$ ceph-deploy calamari connect ospctl1 ospctl2  ospctl3 ceph-osd1 \
ceph-osd2 ceph-osd3
```

Create data pools

```
$sudo ceph osd pool create images 1024
$sudo ceph osd pool create volumes 1024
$sudo ceph osd pool create backups 1024
$sudo ceph osd lspools
```

Create images keyring

```
$ sudo ceph auth get-or-create client.images mon 'allow r' osd 'allow
```

```
class-read object_prefix rbd_children, allow rwx pool=images'
$ sudo ceph auth get-or-create client.images \
> ceph.client.images.keyring
```

Create volumes keyring

```
$ sudo ceph auth get-or-create client.volumes mon 'allow r' osd 'allow
class-read object_prefix rbd_children, allow rwx pool=volumes, allow rx
pool=images'
$ sudo ceph auth get-or-create client.volumes \
> ceph.client.volumes.keyring
```

Create backups keyring

```
$ ceph auth get-or-create client.backups mon 'allow r' osd 'allow class-read
object_prefix rbd_children, allow rwx pool=backups'
$ sudo ceph auth get-or-create client.backups \
> ceph.client.backups.keyring
```

Gather and Deoloy ceph.conf

```
$ ceph-deploy --overwrite-conf config pull ospctl1
$ ceph-deploy --overwrite-conf config push ospctl1 ospctl2 ospctl3 \
ospcomp1 ospcomp2 ceph-osd1 ceph-osd2 ceph-osd3
```

Gather and deploy new client keys

```
$ ssh ospctl1 cat /etc/ceph/ceph.client.backups.keyring > \
ceph.client.backups.keyring
$ ssh ospctl1 cat /etc/ceph/ceph.client.images.keyring > \
ceph.client.images.keyring
$ ssh ospctl1 cat /etc/ceph/ceph.client.volumes.keyring > \
ceph.client.volumes.keyring
$ scp ceph.client.backups.keyring ceph.client.images.keyring \
 ceph.client.volumes.keyring ospctl2:/etc/ceph
$ scp ceph.client.backups.keyring ceph.client.images.keyring \
 ceph.client.volumes.keyring ospctl3:/etc/ceph
$ scp ceph.client.backups.keyring ceph.client.images.keyring \
 ceph.client.volumes.keyring ospcomp1:/etc/ceph
$ scp ceph.client.backups.keyring ceph.client.images.keyring \
 ceph.client.volumes.keyring ospcomp2:/etc/ceph
```

Copy libvirt secret files

```
$ cat ceph.client.volumes.keyring | grep key | awk ‘print $3’ | tee \
client.volumes.key
$ vi secret.xml
$ scp secret.xml client.volumes.key ospcomp1:
$ scp secret.xml client.volumes.key ospcomp2:
```

Add libvirt secret files

```
$ sudo virsh secret-define --file secret.xml
$ sudo virsh secret-set-value --secret 09b647f2-c00e-4d80-8506-e7d8053fbba9
--base64 `cat client.volumes.key`
$ sudo virsh secret-list
```

Test Integration

```
# glance image-create --name rhel65u --is-public true --disk-format qcow2 \
 --container-format bare –file \
 /pub/..../rhel-guest-image-6-6.5-20131115.0-1.qcow2.unlock
# glance image-list
# nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
# nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
# nova boot --flavor 2 --image rhel65u --key-name demokp inst1
# nova boot --flavor 2 --image rhel65u --key-name demokp inst2
# nova list
# nova floating-ip-create
# nova add-floating-ip inst1 10.19.139.217
# ssh -i demokp.pem cloud-user@10.19.139.217
# cinder create --display-name test 1
# nova volume-list
# nova volume-attach inst1 996a242c-ebaf-4265-a76a-e20da812de45 auto
# ssh -i demokp.pem cloud-user@10.19.139.217
# nova show inst2
# ssh -i demokp.pem cloud-user@10.19.139.217 ping -c 3 10.0.1.5
# cinder create --display-name test2 1
# cinder backup-create 1488a3cd-5083-40a5-a189-969eb980576e --display-name
# volume-backup --display-description "A volume backup"
# cinder backup-list
```

Show Pool usage

```
# ceph osd lspools
# rados df -p images
# rados df -p volumes
# rados df -p backups
```

Appendix C: Example Configuration Files

/etc/hosts on ICE Admin host

```
10.19.139.71 ospctl1-storage
10.19.139.72 ospctl2-storage
10.19.139.73 ospctl3-storage
```

/etc/hosts on all other hosts in this document

```
172.31.139.71 ospctl1-storage
172.31.139.72 ospctl2-storage
172.31.139.73 ospctl3-storage
172.31.139.74 ospcomp1-storage
172.31.139.75 ospcomp2-storage
172.31.141.55 ceph-osd1-storage
172.31.141.56 ceph-osd2-storage
172.31.141.57 ceph-osd3-storage
```

Example cephdeploy.conf

```
# This file was automatically generated after ice_setup.py was run. It
provides
```

```
# the repository url and GPG information so that ceph-deploy can install the
# repositories in remote hosts.
#

# ceph-deploy subcommands

[ceph-deploy-calamari]
master = ice-admin.cloud.lab.eng.bos.redhat.com


# Repositories

[calamari-minion]
name=Calamari
baseurl=http://ice-admin.cloud.lab.eng.bos.redhat.com/static/calamari-minion
s
gpgkey=http://ice-admin.cloud.lab.eng.bos.redhat.com/static/calamari-minions
/release.asc
enabled=1
proxy=_none_

[ceph]
name=Ceph
baseurl=http://ice-admin.cloud.lab.eng.bos.redhat.com/static/ceph/0.80.4-1.g
67b5193
gpgkey=http://ice-admin.cloud.lab.eng.bos.redhat.com/static/ceph/0.80.4-1.g6
7b5193/release.asc
default=true
proxy=_none_
```

Example /etc/ceph/ceph.conf

```
[[global]
fsid = 38....6f
mon_initial_members = ospctl1-storage, ospctl2-storage, ospctl3-storage
mon_host = 172.31.139.71,172.31.139.72,172.31.139.73
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
filestore_xattr_use_omap = true
public_network = 172.31.0.0/16
cluster_network = 172.30.0.0/16
osd_pool_default_size = 2

[client.images]
      keyring = /etc/ceph/ceph.client.images.keyring
[client.volumes]
      keyring = /etc/ceph/ceph.client.volumes.keyring
[client.backups]
      keyring = /etc/ceph/ceph.client.backups.keyring
```

Example ceph.client.images.keyring

```
[client.images]
      key = AQ....==
```

```
      caps mon = "allow r"
      caps osd = "allow class-read object_prefix rbd_children, allow rwx
pool=images"
```

Example ceph.client.volumes.keyring

```
[client.volumes]
      key = AQ....==
      caps mon = "allow r"
      caps osd = "allow class-read object_prefix rbd_children, allow rwx
pool=volumes, allow rx pool=images"
```

Example ceph.client.backups.keyring

```
[client.backups]
      key = AQ....==
      caps mon = "allow r"
      caps osd = "allow class-read object_prefix rbd_children, allow rwx
pool=backups"
```

Example storage node /etc/sysconfig/iptables

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 6800:6840 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

Example controller node /etc/sysconfig/iptables minus OSP related rules

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [12:1124]
### Skipped OSP rules ##
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 6789 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```