Red Hat Reference Architecture Series

High Availability

with Red Hat Enterprise Linux OpenStack Platform 4

Jacob Liberman, RHCE

Version 1.0

June, 2014

redhat.

1801 Varsity Drive™
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

OpenStack is a U.S. registered trademark of the OpenStack Foundation.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo and Xeon are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2014 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).

The information contained herein is subject to change without notice. Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc.

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

Send feedback to refarch-feedback@redhat.com

Table of Contents

1 Executive Summary

Red Hat Enterprise Linux OpenStack Platform (RHEL OSP) 4 is a reliable cloud infrastructure platform based on OpenStack. RHEL OSP 4 has a modular service-based architecture designed for massive scalability. Some OpenStack services scale natively. Others require a network load balancer to scale. This document describes Red Hat's approach to scaling OpenStack by integrating RHEL OSP 4 with the Red Hat Enterprise Linux High-Availability (RHEL -HA) Add-On.

The general approach is that all OpenStack services are managed and monitored in isolated RHEL-HA clusters. A high availability network load balancer provides access to the service endpoints. There are no direct connections from the clients to the services.

Benefits of this approach include:

- Individual services can be scaled as needed
- Service isolation simplifies troubleshooting
- Granular startup control satisfies service interdependencies
- Highly available state database and messaging queues

The first section of this document describes the RHEL OSP 4 HA architecture. The second section describes the lab environment, hardware, and software versions used to test the reference architecture. The third section lists the steps performed by the Red Hat Systems Engineering team to deploy the reference architecture in their lab using the same software customers can download from Red Hat. The steps were performed without using an installer in order to expose the underlying architecture and design decisions often masked by an installer.

2 Architecture Overview

This section describes the software components used to develop this reference architecture.

2.1 OpenStack Platform

Red Hat Enterprise Linux OpenStack Platform provides the foundation to build private or public Infrastructure-as-a-Service (IaaS) for cloud-enabled workloads. It allows organizations to leverage OpenStack, the largest and fastest growing open source cloud infrastructure project, while maintaining the security, stability, and enterprise readiness of a platform built on Red Hat Enterprise Linux.

Red Hat Enterprise Linux OpenStack Platform gives organizations a truly open framework for hosting cloud workloads, delivered by Red Hat subscription for maximum flexibility and cost effectiveness. In conjunction with other Red Hat technologies, Red Hat Enterprise Linux OpenStack Platform allows organizations to move from traditional workloads to cloud-enabled workloads on their own terms and timelines, as their applications require. Red Hat frees organizations from proprietary lock-in, and allows them to move to open technologies while maintaining their existing infrastructure investments.

Unlike other OpenStack distributions, Red Hat Enterprise Linux OpenStack Platform provides a certified ecosystem of hardware, software, and services, an enterprise lifecycle that extends the community OpenStack release cycle, and award-winning Red Hat support on both the OpenStack modules and their underlying Linux dependencies. Red Hat delivers long-term commitment and value from a proven enterprise software partner so organizations can take advantage of the fast pace of OpenStack development without risking the stability and supportability of their production environments.

2.1.1 Red Hat Enterprise Linux OpenStack Platform 4

Red Hat Enterprise Linux OpenStack Platform 4 is based on the upstream “Havana” OpenStack release. Red Hat Enterprise Linux OpenStack Platform 4 is Red Hat’s fourth release. The first release was based on the “Essex” OpenStack release. Red Hat’s second release was based on the “Folsom” OpenStack release. It was the first release to include extensible block and volume storage services. The Havana release included all of Folsom’s features along with a more robust network automation platform. RHEL OSP 4 includes all of Havana’s features along with support for metering and orchestration, support for the Foreman Puppet-based installer, and the inclusion of Red Hat’s High Availability and Load Balancer Enterprise Linux Add-ons.

2.1.2 OpenStack Terminology

OpenStack is a complicated project comprised of many modular services. Like OpenStack, the terminology used to describe OpenStack evolves over time. This section defines the terminology and naming conventions used in this document.

2.1.2.1 Server Roles

A typical OpenStack deployment consists of multiple servers performing various roles. Some roles are performed by dedicated servers. Other roles might overlap across the same servers.

- **Controller Node** – The OpenStack controller nodes run the API services, schedulers, and auxiliary services such as the state database, object caching system, and messaging system. Each OpenStack service can be distributed across all or a subset of the controller nodes.

  There are three dedicated controller nodes in this reference architecture. All OpenStack API and scheduler services run on the controller nodes including the Neutron server and agents. The MySQL state database resides on an iSCSI target accessible by all three controller nodes. The controller nodes also run Memcached for object caching and Qpid for messaging. The controller nodes are made highly available with Pacemaker/Corosync. Individual services run active/active, active/passive, or active/hot-standby as dictated by the capabilities and constraints of each service.

- **Load Balancer** – OpenStack Havana has a Load Balancer-as-a-Service (LbaaS) feature. However, in this reference architecture, Load Balancer refers to a specific role. The load balancer servers run HAproxy to load balance TCP/HTTP connection requests across two or more controller nodes. The load balancers act as an entry point to the OpenStack high availability cluster. Clients never connect directly to the servers behind the load balancers. They always connect through the load balancers.

  In this reference architecture, two dedicated servers perform this role. They are Red Hat Enterprise Linux servers running HAproxy. They are made highly available via Pacemaker. It is possible to run HAproxy directly on the controller nodes at the cost of added complexity. It is also possible to replace the Haproxy-based load balancers with alternate hardware or software load balancers.

- **Back end Storage Server** – Many OpenStack services either use or provide storage. Virtual machines are allocated ephemeral storage by default and may also require persistent block storage. Other services may use object storage for scalability or availability. In this reference architecture, the Back end Storage Server refers to the server or servers that provide the underlying storage for the OpenStack storage services. Back end storage servers might take many forms including dedicated NAS devices or network file system shares.

  In this reference architecture a dedicated NFS server acts as the Back End Storage server. The Cinder and Glance services both use the Back End Storage server. The NFS server used in the Systems Engineering lab was not made highly available. The steps and software required to make NFS highly available are not described in this document as they are well described elsewhere.

- **Network Node** – the servers in this role provide user-defined networking capability to the OpenStack tenants. These services include virtual bridging, DHCP servers, and virtual routing. Network nodes might also run the API and scheduler services for Neutron. The Network Nodes can be dedicated servers or co-resident with the Controller Nodes.

In this reference architecture the Network Node role is performed by the three Controller Nodes.

- **Compute Node** – Compute nodes provide the processing, memory, and networking resources required to run instances. In RHEL OSP 4, the compute nodes run the KVM hypervisor. KVM manages virtual machine access to the underlying hardware.

  OpenStack compute nodes scale natively. New Compute nodes register themselves with the Nova scheduler. The scheduler load balances new instances across all registered Compute nodes in a round robin fashion. Therefore, an external load balancer is not required for the Compute nodes. There are three Compute nodes in this reference architecture. Each compute node runs as a single node cluster. Pacemaker controls monitoring and restarting the local Compute node in the event of a service interruption.

2.1.2.2 Network Names

A typical OpenStack deployment includes several network roles. In some cases the roles overlap across the same physical interfaces or switches. In others each role has a dedicate network interface and switches. This section of the paper describes the roles used in this reference architecture.

- **Management network** – the Management network is used to perform system maintenance tasks such as installing software. In a private cloud scenario, internal users might access the cloud infrastructure via the management network.

  In this reference architecture the Management network serves two purposes. First, the load balancers listen for connection requests on this network. Each service is associated with a virtual IP address on the management network. The load balancers forward connection requests to the API listeners on the controller nodes. Second, the management network is used to install software and transfer files between the OpenStack servers. The management network is instantiated on the *em1* interface with IP v4 IP addresses in the 10.19.137.0/24 ranges.

- **Service network** – this network exposes the OpenStack APIs to the load balancers. It also handles inter-service communication between the OpenStack services and schedulers.

  In this reference architecture the *em2* interface acts as the service network. The interfaces are assigned IP v4 addresses in the 172.16.2.0/24 range.

- **Tenant networks** – virtual machines communicate over this network within the cloud deployment. The addressing requirements of this network depend on the plugin that is used.

  In this reference architecture the 10 GB *p3p1* interface is used for the data network. The interfaces are tagged with VLANs 1000-1010 for tenant communication. The interfaces do not have IP addresses.

- **Storage Network** – this network is dedicated for storage traffic between the OpenStack servers and the back end storage node.

  In this reference architecture the 10 GB *p3p2* interface is the storage network. The

interfaces are assigned with IP v4 IP addresses in the 172.31.0.0/16 range.

2.1.3 High Availability OpenStack Services in this Document

This section introduces the OpenStack services covered in this document.

2.1.3.1 Identity Service ("Keystone")

This is a central authentication and authorization mechanism for all OpenStack users and services. It supports multiple forms of authentication including standard username and password credentials, token-based systems and AWS-style logins that use public/private key pairs. It can also integrate with existing directory services such as LDAP.

The Identity service catalog lists all of the services deployed in an OpenStack cloud and manages authentication for them through *endpoints*. An endpoint is a network address where a service listens for requests. The Identity service provides each OpenStack service -- such as Image, Compute, or Block Storage -- with one or more endpoints.

The Identity service uses *tenants* to isolate resources. By default users in one tenant can't access resources in another even if they reside within the same OpenStack cloud deployment or physical host. The Identity service issues tokens to authenticated users. The endpoints validate the token before allowing user access. User accounts are associated with *roles* that define their access credentials.

2.1.3.2 Image Service ("Glance")

This service registers and delivers virtual machine images. They can be copied via snapshot and immediately stored as the basis for new instance deployments. Stored images allow OpenStack users and administrators to provision multiple servers quickly and consistently. The Image Service API provides a standard RESTful interface for querying information about the images.

By default the Image Service stores images in the */var/lib/glance/images* directory of the local server's filesystem where Glance is installed. The Glance API can also be configured to cache images in order to reduce image staging time. The Image Service supports multiple back end storage technologies including Swift (the OpenStack Object Storage service), Amazon S3, Ceph, and Red Hat Storage Server.

2.1.3.3 Compute Service ("Nova")

OpenStack Compute provisions and manages virtual machines. It is the backbone of OpenStack's IaaS functionality. OpenStack Compute scales horizontally on standard hardware enabling the favorable economics of cloud computing.

Key features of OpenStack Compute include:

- Distributed and asynchronous architecture, allowing scale out fault tolerance for virtual machine instance management
- Management of commoditized virtual server resources, where predefined virtual hardware profiles for guests can be assigned to new instances at launch

- VNC access to instances via web browsers

2.1.3.4 Block Storage ("Cinder")

While the OpenStack Compute service provisions ephemeral storage for deployed instances based on their hardware profiles, 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. This service's features include:

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

2.1.3.5 Network Service ("Neutron")

OpenStack Networking is a scalable API-driven service for managing networks and IP addresses. OpenStack Networking gives users self-service control over their network configurations. Users can define, separate, and join networks on demand. This allows for flexible network models that can be adapted to fit the requirements of different applications. OpenStack Networking has a pluggable architecture that supports numerous virtual networking technologies as well as native Linux networking mechanisms including **`openvswitch`** and **`linuxbridge`**.

2.1.3.6 Dashboard ("Horizon")

The OpenStack Dashboard is an extensible web-based application for provisioning and controlling compute, storage, and networking resources.

2.1.4 OpenStack Services Not Described in this Document

This section describes the OpenStack services supported in RHEL OSP 4 but not made highly available in this document.

2.1.4.1 Orchestration ("Heat")

Heat is an OpenStack orchestration engine. It can launch multiple composite cloud applications based on text-based template files. The templates can describe infrastructure resources including servers, floating IP addresses, storage, security groups, and users.

2.1.4.2 Telemetry ("Ceilometer")

Ceilometer provides infrastructure to collect measurements within OpenStack. It is primarily useful for monitoring and metering. Most services have a Ceilometer plugin. It is centralized so no two agents need to be written to collect the same data.

2.1.4.3 Object Storage (“Swift”)

Swift i a highly available distributed object store. The Swift architecture is generally comprised of several servers with unique roles. These include the proxy server, object servers, and container servers.

2.2 High Availability with HAproxy and pacemaker

This section describes the high availability technologies used in this reference architecture along with the rationale for their inclusion.

2.2.1.1 HAProxy

HAProxy is a TCP/HTTP load balancer. In this reference architecture HAProxy load balances connection requests to many OpenStack services. HAProxy supports a number of load balancing schemes including active/active and active/passive.

2.2.1.2 Pacemaker

Pacemaker is an open-source high availability resource manager. It detects and recovers from machine and application-level failures. In this reference architecture pacemaker restarts OpenStack services if they fail. It also manages the state database, messaging system, and HAProxy load balancers.

2.2.2 Highly Available Resources

Although the same infrastructure software is used to cluster and load balance all of the services in this reference architecture, the high availability policies and software placement vary on a per service basis. This section describes the high availability resources and their policies. It also lists start up order dependencies between services.

Conceptually each service operates as an independent HA cluster on dedicated servers. With this approach each service could be run on separate servers or virtual machines and scaled as needed. **Figure 2.2.2.1: Segregated Highly Available Clusters**depicts the RHEL OSP 4 architecture from a conceptual standpoint.

The only relationships that must be enforced are the start-up dependencies and network connections. **Figure 2.2.3.1: Service Start Order and Dependencies** depicts the start-up dependencies.

Each figure includes a legend that identifies the type of high availability offered by each resource.

- **Active/active** refers to the approach where all nodes run the service simultaneously. Connection requests are forwarded to any of the nodes that own the resource. If a node fails, the others continue to service requests.
- **Active/passive** refers to the approach where a single node runs the service. All connection requests are routed to the active node. In the event of a node failure, a passive node is activated. Subsequent connection requests are routed to the passive node.

- **Hot standby** refers to the approach where all nodes simultaneously run the service. Only a single node fields the connection requests. In the event that the active node fails, new connection requests are seamlessly routed to a hot standby node.
- **Mixed** refers to a resource that utilizes a variety of the previously described approached for one or more services. Mixed varies on a case by case basis. Generally mixed high availability includes some services in Active/Active mode and others in Active/Passive or Hot standby.
- **Single node** refers to services running on only one node. These standalone services may be managed by Pacemaker which restarts them if it detects a failure.

2.2.2.1 State Database

A state database resides at the heart of an OpenStack deployment. This SQL database stores most of the build-time and run-time state information for the cloud infrastructure including available instance types, networks, and the state of running instances in the compute fabric. Although OpenStack theoretically supports any SQL-Alchemy compliant database, RHEL OSP 4 uses MySQL, a widely used open source database packaged with Red Hat Enterprise Linux 6.

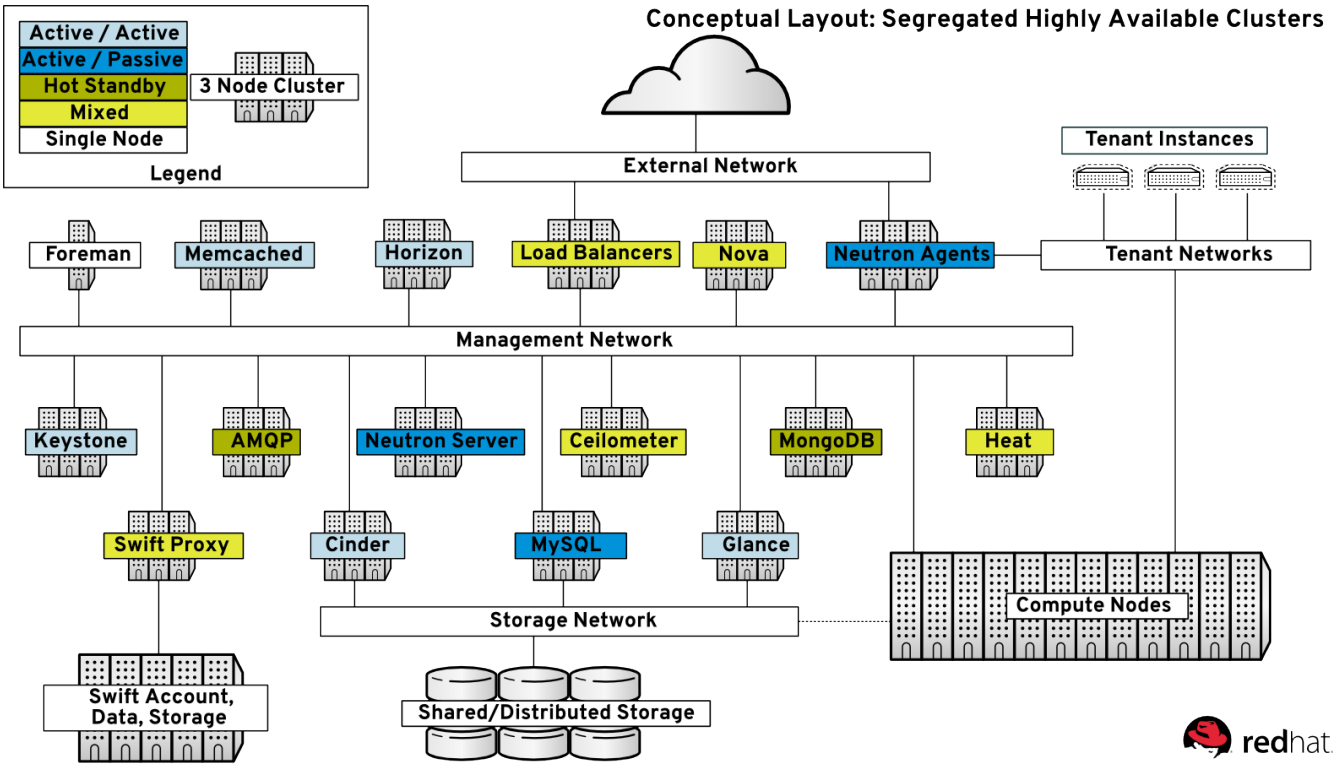

Figure 2.2.2.1: Segregated Highly Available Clusters

2.2.2.2 Messaging

Enterprise messaging systems let programs communicate by exchanging messages. OpenStack services use enterprise messaging to communicate tasks and state changes between endpoints, schedulers, services and agents. RHEL OSP 4 uses Qpid for open source enterprise messaging. Qpid is an Advanced Message Queuing Protocol (AMQP) compliant, cross-platform enterprise messaging system developed for low latency based on an open standard for enterprise messaging. In this reference architecture Qpid runs in hot standby mode using a stick table. All controller nodes run Qpid. But load balancer rules route subsequent connections to the messaging server that fielded the initial request.

2.2.2.3 Memcached

Memcached is an in-memory key-value store for arbitrary data. It is intended to speed up applications by alleviating database load. In RHEL OSP 4, Memcached runs on the high availability controllers in active/active mode.

2.2.2.4 OpenStack Services

Figure 2.2.2.2: High Availability Deployment depicts the OpenStack service placement in relation to the physical servers and networks.

- **Keystone** – runs as a cloned (active/active) service on the HA cluster of three controllers

- **Glance** -- runs as a cloned (active/active) service on the highly available controllers. This includes the NFS Glance file system mounted simultaneously to all controllers, the **`glance-api`**, and the **`glance-registry`** service.
- **Nova** -- runs as mixed active/passive and active/active services on the HA controller nodes. At the time of writing, the **`nova-scheduler`** service runs in active/passive mode. The **`nova-api`**, **`nova-conductor`**, **`nova-consoleauth`**, and **`nova-novncproxy`** run as cloned (active/active) resources.
- **Cinder** -- runs as a cloned (active/active) service on the HA controllers. The high availability services include the **`cinder-scheduler`**, **`cinder-api`**, and **`cinder-volume`** services.
- **Neutron** – the Neutron server runs as an active/passive HA cluster resources.
- **Neutron agents** – the Neutron agents – L3, DHCP, Open vSwitch, and Metadata agents run as active/passive HA cluster resources on the controller nodes.
- **Horizon** – The OpenStack dashboard runs as a cloned (active/active) **`httpd`** resources on the controller nodes.
- **Nova-compute** – This service runs as a single node cluster resource on each compute node. Pacemaker monitors the **`nova-compute`** service and fences or restarts the service in the event an interruption is detected.
- **AMQP** – runs as a hot-standby service on all controller nodes. The services runs on all controllers but a stick table setting on the load balancers directs all connections to a single running instance as long as it is available.
- **MySQL** – the MySQL server runs as an active/passive resource group on the controller nodes. One controller mounts the file system, activates the logical volume, and starts the MySQL server at one time. The load balancer sends connections to the active MySQL node in preparation for the inclusion of future active/active databases.
- **Memcached** – runs as a cloned (active/active) HA cluster resource

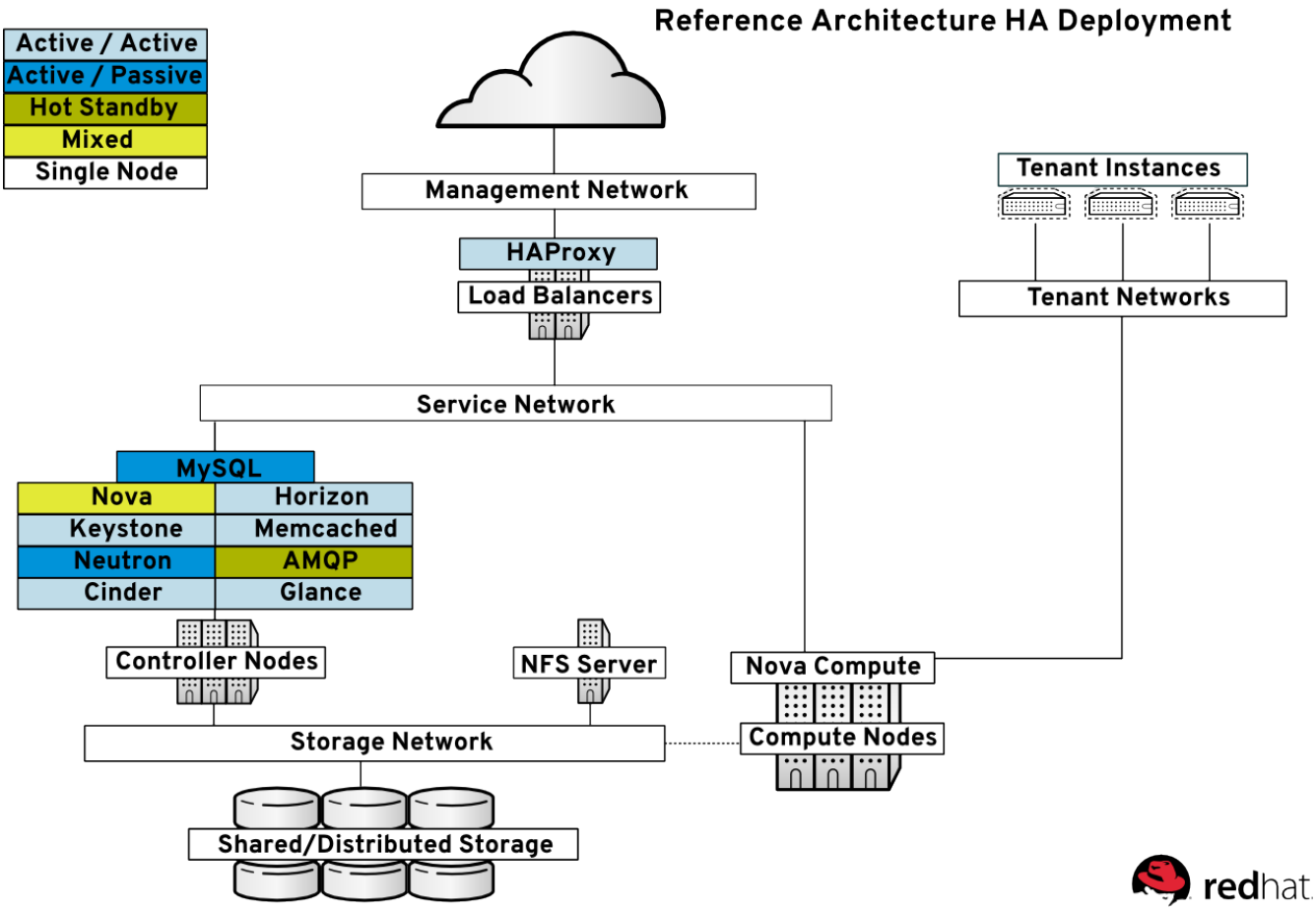

Figure 2.2.2.2: High Availability Deployment

2.2.3 Start up Dependencies

Figure 2.2.3.1: Service Start Order and Dependencies depicts the start up dependencies between HA services. At the lowest level, the load balancers must be online first. Supporting services including AMQP for messaging, MongoDB for Ceilometer, and MySQL for the state database must follow. Next, Keystone and Memcached may start in parallel. The core OpenStack services follow. Finally, the services required by the compute infrastructure may be brought online once all supporting services are in place.

The checkpoint layers depicted in the graphic refer to periods of inactivity. These are designed to allow services to settle before proceeding to the next order of dependencies. The checkpoints can be of variable length. What is important is that the services preceding the checkpoint start without error on all HA cluster nodes before the next order of dependencies are brought online.

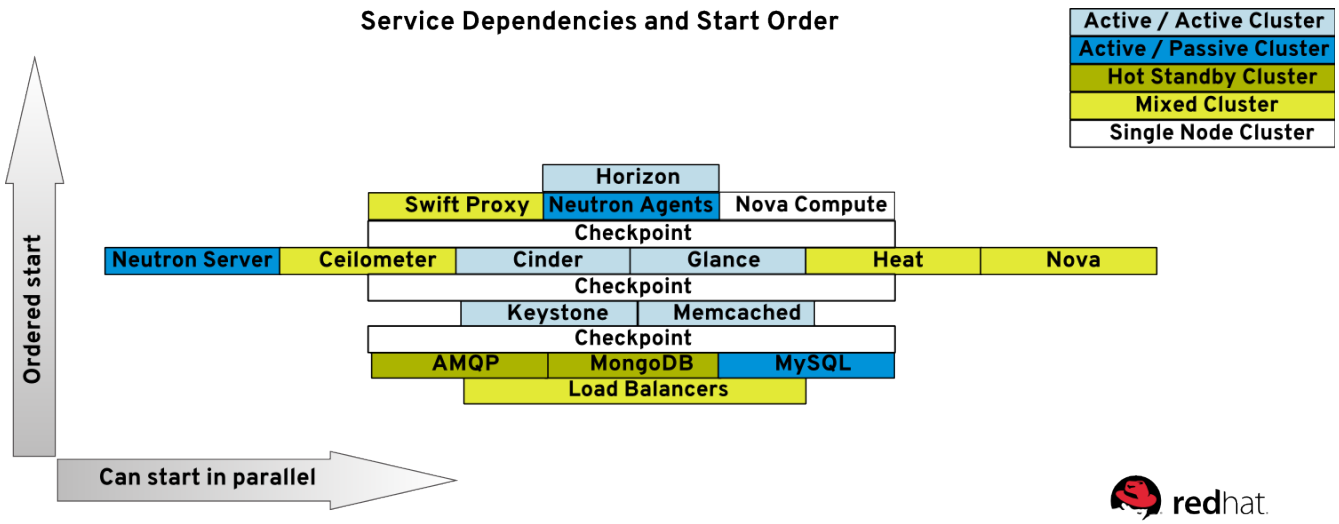

Figure 2.2.3.1: Service Start Order and Dependencies

2.3 Service Placement

Table 2.3.1: Service Placement shows the final service placement for all OpenStack services. The controller nodes run the majority of services. The active/active services run on all controller nodes. The active/passive services are distributed across the controllers. The compute nodes run the compute service. The dedicated load balancers run HAproxy and the service virtual IP addresses.

Host	Role	Service	Type
rhos[0,1]	Load Balancer	haproxy	Clone set
		vip-mysql	Single server
		vip-qpid	Single server
		vip-keystone	Single server
		vip-glance	Single server
		vip-cinder	Single server
		vip-neutron	Single server
		vip-nova	Single server
		vip-horizon	Single server
rhos[2-4]	Controller	qpidd	Clone set
		mysql-ha-lvm	Single server
		mysql-fs	Single server
		mysql-db	Single server
		keystone	Clone set

		memcached	Clone set
		glance-fs	Clone set
		glance-registry	Clone set
		glance-api	Clone set
		cinder-api	Clone set
		cinder-scheduler	Clone set
		cinder-volume	Clone set
		openvswitch	Single server
		neutron-server	Single server
		neutron-openvswitch-agent	Single server
		neutron-dhcp-agent	Single server
		neutron-l3-agent	Single server
		neutron-metadata-agent	Single server
		nova-consoleauth	Clone set
		nova-novncproxy	Clone set
		nova-api	Clone set
		nova-scheduler	Single server
		nova-conductor	Clone set
		httpd	Clone set
rhos[5-7]	Compute nodes	openvswitch	Single server
		neutron-db-check	Single server
		neutron-openvswitch-agent	Single server
		messagebus	Single server
		libvirtd	Single server
		nova-compute	Single server

Table 2.3.1: Service Placement

3 Reference Architecture Configuration Details

This section of the paper describes the hardware, software, and procedures used to configure this reference architecture in the lab. Best practices learned in the lab are shared throughout this document.

3.1 Environment

The reference architecture environment consists of the components required to build a small Red Hat Enterprise Linux OpenStack Platform cloud infrastructure.

3.1.1 Network Topology

Figure 3.1.1.1 Network Topology shows the network topology of this reference architecture.

- All eight servers communicate via the lab network switch on the 10.19.136.0/21 network. The load balancers listen for API service requests on this network. It is also used to install software and otherwise manage the servers. This topology mimics a private cloud where client and management communication do not need to be segmented for security reasons.
- The tenant network carries communication between virtual machines and software-defined networking components. It is the private network over which the instances communicate. In this reference architecture, a network switch connected to 10 GB interfaces on the compute nodes is tagged to VLAN IDs 1000:1010.

  NOTE: The tenant network carries tenant network traffic over tagged VLANs. The interfaces connected to this network are not configured with IPv4 addresses by the OpenStack administrator. Instead, instances and services are allocated addresses within user-created subnets on tenant networks. Network namespaces prevent different users' subnets from conflicting with each other or with the infrastructure's own subnets.
- All servers communicate with shared storage via a second 10Gb storage network switch on the 172.31.0.0/16 network. This network delivers the Image service virtual machine disk images as well as the persistent block storage for the Block Storage service via NFS. It also provides the iSCSI shared storage for the high availability MySQL state database.
- The Service network carries service communication between the load balancers and the service listeners. These include the various schedulers and agents deployed in the OpenStack environment. The service traffic is segmented from the tenant and management traffic. In this reference architecture, the second 1GB interface on the 172.16.2.0/24 network is the service network.

This reference architecture uses four physical networks. However it is possible to deploy

supported OpenStack solutions with more or fewer networks.

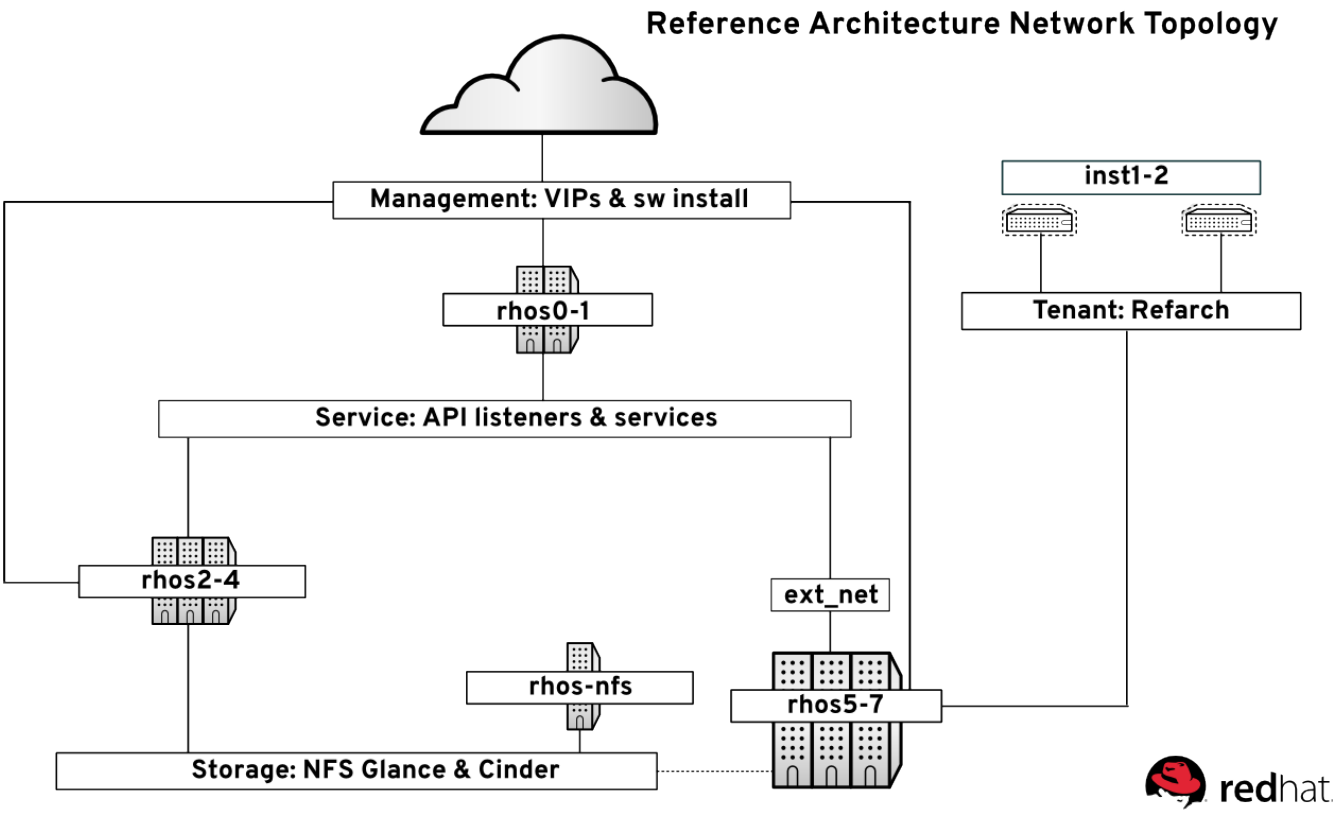

Figure 3.1.1.1: Network Topology

3.1.2 Network Addresses

Table 3.1.2.1: Host IP Addresses lists the IPv4 Addresses used in this reference architecture by server host name and role. All servers have interfaces on four networks: management, service, tenant, and storage. There are three roles: load balancer, controller, or compute node. A minimum of two load balancers, three controllers, and one compute node are required to implement this reference architecture. In this example three compute nodes were used.

NOTE: The data network carries virtual machine and software-defined network device traffic over tagged VLANs. The interfaces connected to this network are not configured with IPv4 addresses by the OpenStack administrator. Instead they act as bridges to software-defined network devices in network name spaces.

Host	Role	Network	Interface	Network Address
rhos0	Load Balancer	Mgmt	em1	10.19.137.100
		Storage	p3p2	172.31.139.100
		Service	em2	172.16.2.100

rhos1	Load Balancer	Mgmt	em1	10.19.137.101
		Storage	p3p2	172.31.139.101
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.101
rhos2	Controller	Mgmt	em1	10.19.137.102
		Storage	p3p2	172.31.139.102
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.102
rhos3	Controller	Mgmt	em1	10.19.137.103
		Storage	p3p2	172.31.139.103
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.103
rhos4	Controller	Mgmt	em1	10.19.137.104
		Storage	p3p2	172.31.139.104
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.104
rhos5	Compute	Mgmt	em1	10.19.137.105
		Storage	p3p2	172.31.139.105
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.105
rhos6	Compute	Mgmt	em1	10.19.137.106
		Storage	p3p2	172.31.139.106
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.106
rhos7	Compute	Mgmt	em1	10.19.137.107
		Storage	p3p2	172.31.139.107
		Tenant	p3p1	VLAN 1000:1010
		Service	em2	172.16.2.107

Table 3.1.2.1: Host IP Addresses

3.2 Software Details

This section of the reference architecture lists the required software revisions. It also lists software configuration details related to security including **`SELinux`** and **`iptables`**. Customers who use the correct OpenStack and Red Hat Enterprise Linux channels on Red

Hat Network (RHN) or Subscription Manager meet the minimum required software versions.

3.2.1 Required Channels

Red Hat Enterprise Linux OpenStack Platform is available via the Red Hat Network channels and RHN Certificate Server repositories listed in **Table 3.2.1.1: Required Channels.**

Channel	Source
rhel-x86_64-server-6	RHN Classic
rhel-x86_64-server-6-ost-3	RHN Classic
rhel-x86_64-server-ha-6	RHN Classic
rhel-x86_64-server-6-mrg-messaging-2	RHN Classic
rhel-x86_64-server-lb-6	RHN Classic
rhel-6-server-rpms	RHN Certificate
rhel-6-server-openstack-4.0-rpms	RHN Certificate
rhel-6-server-rh-common-rpms	RHN Certificate
rhel-ha-for-rhel-6-server-rpms	RHN Certificate
rhel-lb-for-rhel-6-server-rpms	RHN Certificate
• rhel-6-server-mrg-messaging-2-rpms	RHN Certificate

Table 3.2.1.1: Required Channels

NOTE: This reference architecture uses RHN Classic in all examples via a lab satellite server.

3.2.2 Software Versions

Table 3.2.2.1: Software Versions lists the software versions used to develop this reference architecture.

Host	Software	Version
Load Balancer	pacemaker	1.1.10-14
	fence-agents	3.1.5-35
	resource-agents	3.9.2-40
	pcs	0.9.90-2
	cman	3.0.12.1-59
	ccs	0.16.2-69.el6_5.1
	haproxy	1.4.24-2
Compute Nodes	openstack-nova-compute	2013.2.3-7
	openstack-utils	2013.2-3

	python-cinder	2013.2.3-2
	openstack-neutron-openvswitch	2013.2.3-7
	pacemaker	1.1.10-14.el6_5.3
	fence-agents	3.1.5-35.el6_5.4
	resource-agents	3.9.2-40.el6_5.7
	pcs	0.9.90-2.el6_5.3
	cman	3.0.12.1-59.el6_5.2
	ccs	0.16.2-69.el6_5.1
Controller	ccs	0.16.2
	cman	3.0.12.1
	fence-agents	3.1.5
	iproute	2.6.32
	memcached	1.4.4
	mysql-server	5.1.73
	openstack-cinder	2013.2.3
	openstack-dashboard	2013.2.3
	openstack-glance	2013.2.3
	openstack-keystone	2013.2.3
	openstack-neutron	2013.2.3
	openstack-neutron-openvswitch	2013.2.3
	openstack-nova-api	2013.2.3
	openstack-nova-conductor	2013.2.3
	openstack-nova-console	2013.2.3
	openstack-nova-novncproxy	2013.2.3
	openstack-nova-scheduler	2013.2.3
	openstack-selinux	0.1.3
	pacemaker	1.1.10
	pcs	0.9.90
	qpid-cpp-server	0.18
	qpid-tools	0.18
	resource-agents	3.9.2

Table 3.2.2.1: Software Versions

3.3 Security Reference

This section describes the security configuration used in this reference architecture.

3.3.1 Firewall Configuration

. **Table 3.3.1.1: Allowed iptables Ports by Role** lists the allowed ports by host, and role.

Port	Host	Role
22, 80, 2049, 3306, 5000, 5404, 5405, 5672, 6080, 6081, 8000, 8003, 8004, 8080, 8774, 8775, 8776, 8777, 9191, 9292, 9696, 11211, 27017, 35357	rhos[0,1]	Load balancer
22, 80, 443 , 2049, 3306, 5000, 5405, 5672, 6080, 6081, 8774, 8775, 8776, 9191, 9292, 9696, 9697, 11211, 35357	rhos[2-4]	Controller
22, 53, 67, 443, 5900:5999	rhos[5-7]	Compute node

Table 3.3.1.1: Allowed iptables Ports by Role

3.3.2 SELinux Configuration

Red Hat Enterprise Linux OpenStack Platform supports **`SELinux`** in **enforcing** mode in Red Hat Enterprise Linux 6.5. **Table 3.3.2.1: Supported SELinux Package Versions** lists the required packages. **Table 3.3.2.2 SELinux Boolean Values** shows the SELinux Boolean values that must be set to **True** by server type.

Package	Version
libselinux	2.0.94
selinux-policy	3.7.19
selinux-policy-targeted	3.7.19
openstack-selinux	0.1.3

Table 3.3.2.1: Supported SELinux Package Versions

Server	Version
Controller	httpd_can_network_connect
Compute	virt_use_nfs

Table 3.3.2.2: SELinux Boolean Values

3.4 Hardware Details

This section describes the hardware that was used to stand up this reference architecture in the Systems Engineering lab.

3.4.1 Server Hardware Configuration

Table 3.4.1.1: Server Hardware lists the hardware specifications for the servers used in this reference architecture. The bottom row lists the specifications for the Red Hat Storage servers. The top row lists the hardware specifications for all remaining OpenStack servers.

> **NOTE:** Red Hat Enterprise Linux OpenStack Platform servers do not require identical hardware. The hardware used in this reference architecture meets the minimum requirements outlined in the OpenStack documentation. The **Red Hat Enterprise Linux OpenStack Platform 4** installation guide contains further details.

Hardware	Specifications
8 x DELL BladeServer – PowerEdge M520	2 x Intel(R) Xeon(R) CPU E5-2450 0 @ 2.10GHz (8 core)
	2 x Broadcom NetXtreme II BCM57810 10 Gigabit Ethernet 4 x Broadcom NetXtreme BCM5720 Gigabit Ethernet
	8 x DDR3 4096 MB @1600 MHZ DIMMs
	2 x 146GB SAS internal disk drives

Table 3.4.1.1: Server Hardware

3.4.2 Network Hardware and Topology

By convention each of the HA virtual IP addresses are named *rhos-<service name>-vip* and addressed 10.19.137.1-13.

Each server has three interfaces: a management network interface, a private interface for service communication, and an interface devoted to network storage.

The management network interfaces and network storage interfaces are connected via an HP Procurve 5400 switch. The network storage interfaces are connected to a 10GB switch module along with the Dell Equallogic shared storage.

3.4.3 Storage Configuration

Three controller nodes share a single iSCSI target for shared storage. The iSCSI target resides on a Dell Equallogic iSCSI storage appliance. This is described in **Table 3.4.3.1: iSCSI Storage Disk Usage**.

Server	Disk Usage
rhos[2-4]	500 GB

Table 3.4.3.1: iSCSI Storage Disk Usage

4 Implementing High Availability OpenStack

This section describes the process that was followed to stand up HA OpenStack in the Systems Engineering lab.

4.1 Prepare the Hosts

Complete the steps described in this section on all load balancers, controllers, and compute nodes.

4.1.1 Configure Name Resolution

Both Domain Name Service (DNS) and local */etc/hosts* files were used for name resolution. The steps for creating a hosts file are shown.

> **NOTE:** The */etc/hosts* file is not required if forward and reverse DNS is available. In the Systems Engineering lab environment, forward and reverse DNS were available for the management network. The private and storage networks were not available. These addresses were added to the */etc/hosts* file on all nodes. The management network addresses were also added for completeness.

1. Create an */etc/hosts* file that contains IP address to name mappings for all server interfaces and virtual clustered addresses.

```
127.0.0.1        localhost localhost.localdomain localhost4
::1              localhost localhost.localdomain localhost6
10.19.137.1      rhos-mysql-vip
10.19.137.2      rhos-qpid-vip
10.19.137.3      rhos-keystone-vip
10.19.137.4      rhos-memcache-vip
10.19.137.5      rhos-glance-vip
10.19.137.6      rhos-cinder-vip
10.19.137.7      rhos-swift-vip
10.19.137.8      rhos-neutron-vip
10.19.137.9      rhos-nova-vip
10.19.137.10     rhos-horizon-vip
10.19.137.11     rhos-heat-vip
10.19.137.12     rhos-mongo-vip
10.19.137.13     rhos-ceilometer-vip
10.19.137.100    rhos0
172.16.2.100     rhos0-priv
172.31.139.100   rhos0-stor
10.19.137.101    rhos1
172.16.2.101     rhos1-priv
172.31.139.101   rhos1-stor
10.19.137.102    rhos2
172.16.2.102     rhos2-priv
```

```
172.31.139.102   rhos2-stor
10.19.137.103    rhos3
172.16.2.103     rhos3-priv
172.31.139.103   rhos3-stor
10.19.137.104    rhos4
172.16.2.104     rhos4-priv
172.31.139.104   rhos4-stor
10.19.137.105    rhos5
172.16.2.105     rhos5-priv
172.31.139.105   rhos5-stor
10.19.137.106    rhos6
172.16.2.106     rhos6-priv
172.31.139.106   rhos6-stor
10.19.137.107    rhos7
172.16.2.107     rhos7-priv
172.31.139.107   rhos7-stor
172.31.139.110   nfs-server
```

4.1.2 Configure Networks

Configure the IP addresses for all network interfaces and test connectivity. The IP addresses used in this reference architecture are listed in **Table 3.1.2.1: Host IP Addresses**.

4.1.3 Create and Share SSH Keys

SSH keys were created and distributed between lab servers while building this reference architecture in order to share data and execute remote commands.

> **NOTE:** SSH keys are not required to operate OpenStack. They can be deleted after completing the installation.

1. Create an SSH key.

```
[root@rhos0 ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
```

2. Copy the key to all servers.

```
[root@rhos0 ~]# i=0; while [ $i -le 7 ]; do ssh-copy-id -i
/root/.ssh/id_rsa.pub rhos$i; ((i++)); done
Now try logging into the machine, with "ssh 'rhos0'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos1's password: ********
Now try logging into the machine, with "ssh 'rhos1'", and check in:

  .ssh/authorized_keys
```

```
to make sure we haven't added extra keys that you weren't expecting.

root@rhos2's password: ********
Now try logging into the machine, with "ssh 'rhos2'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos3's password: ********
Now try logging into the machine, with "ssh 'rhos3'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos4's password: ********
Now try logging into the machine, with "ssh 'rhos4'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos5's password: ********
Now try logging into the machine, with "ssh 'rhos5'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos6's password: ********
Now try logging into the machine, with "ssh 'rhos6'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.

root@rhos7's password:
Now try logging into the machine, with "ssh 'rhos7'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

3. Test the key by logging into the remote machines.

```
[root@rhos0 ~]# i=0; while [ $i -le 7 ]; do ssh rhos$i uptime; ((i++)); done

 12:57:08 up  1:03,  1 user,  load average: 0.00, 0.00, 0.00
 12:57:08 up  1:03,  1 user,  load average: 0.00, 0.00, 0.00
 12:57:08 up  1:03,  1 user,  load average: 0.07, 0.02, 0.00
 12:57:09 up 45 min,  1 user,  load average: 0.00, 0.00, 0.00
 12:57:09 up  1:03,  1 user,  load average: 0.00, 0.00, 0.00
```

```
12:57:09 up 12 min,  1 user,  load average: 0.01, 0.06, 0.05
12:57:11 up  1:04,  1 user,  load average: 0.00, 0.00, 0.00
12:57:11 up  1:04,  1 user,  load average: 0.00, 0.00, 0.00
```

4. Repeat steps 1-3 on all servers.

4.1.4 Synchronize Time

Synchronize all servers to a central Network Time Protocol (NTP) server. Repeat these steps on all servers. Using an internal NTP server is a good practice.

1. Add a NTP servers to */etc/ntp.conf*.

```
[root@rhos0 ~]# sed --follow-symlinks -i '/server
.*rhel\.pool\.ntp.org/d' /etc/ntp.conf

[root@rhos0 ~]# echo -e "server 10.19.255.2\nserver 10.19.255.3" >>
/etc/ntp.conf

[root@rhos0 ~]# grep -v \# /etc/ntp.conf | grep -v \^\$
driftfile /var/lib/ntp/drift
restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
restrict 127.0.0.1
restrict -6 ::1
includefile /etc/ntp/crypto/pw
keys /etc/ntp/keys
server 10.19.255.2
server 10.19.255.3
```

2. Enable and start **ntpd** and **ntpdate**.

```
[root@rhos0 ~]# chkconfig ntpdate on

[root@rhos0 ~]# chkconfig ntpd on

[root@rhos0 ~]# service ntpdate start
ntpdate: Synchronizing with time server:                   [  OK  ]

[root@rhos0 ~]# service ntpd start
Starting ntpd:                                             [  OK  ]
```

4.1.5 Update the Systems

Register all systems with Red Hat Network or Red Hat CDN and update them to the latest software versions.

1. Red Hat Enterprise Linux OpenStack Platform 4 requires Red Hat Enterprise Linux operating system version 6.5. Version 6.4 is not supported. Verify the operating system version.

```
[root@rhos0 ~]# cat /etc/redhat-release
Red Hat Enterprise Linux Server release 6.5 (Santiago)
```

NOTE: Refer to the **Red Hat Enterprise Linux OpenStack Platform 4 Release Notes** for a complete list of software version requirements.

2. Verify access to the base operating system channel and repository.

```
[root@rhos0 ~]# rhn-channel -l
rhel-x86_64-server-6

[root@rhos0 ~]# yum repolist
Loaded plugins: product-id, refresh-packagekit, rhnplugin, security,
              : subscription-manager
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
This system is receiving updates from RHN Classic or RHN Satellite.
repo id              repo name
status
rhel-x86_64-server-6 Red Hat Enterprise Linux Server (v. 6 for 64-bit x86
12,602
repolist: 12,602
```

3. **Table 3.2.1.1: Required Channels** lists the required channels to implement high availability clustering.

   Verify access to these channels on either Red Hat Network (classic) or the Red Hat Content Delivery Network (CDN.)

4. Update installed software to latest versions.

```
[root@rhos0 ~]# yum update -y
```

NOTE: The output of all **yum** commands has been truncated or removed for brevity.

4.1.6 Disable ACPID

Disable and stop unused services.

```
[root@rhos0 ~]# chkconfig acpid off

[root@rhos0 ~]# service acpid stop
Stopping acpi daemon:                                      [  OK  ]
```

4.1.7 Mount Shared Storage

An NFS server provides the backend storage for this reference architecture. Create a common mount point and mount the NFS share on all servers.

NOTE: Appendix B: Configure Storage Server contains instructions for installing the NFS server.

1. Create a mount point named */srv* on all servers.

```
[root@rhos0 ~]# mkdir /srv
```

2. Update */etc/fstab* to mount the remote NFS share automatically at boot.

```
[root@rhos0 ~]# echo "nfs-server:/srv /srv nfs defaults 0 0" >>
/etc/fstab
```

3. Allow TCP port 2049.

```
[root@rhos0 ~]# iptables -I INPUT -p tcp --dport 2049 -j ACCEPT
```

4. Save and restart **`iptables`** to verify the port is allowed.

```
[root@rhos0 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos0 ~]#         service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos0 ~]# iptables -L -n | grep 2049
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:2049
```

5. Mount the remote NFS share.

```
[root@rhos0 ~]# mount -av | grep srv
nfs-server:/srv on /srv type nfs (rw)
```

4.2 Configure the Load Balancers

OpenStack scales horizontally to suit the cloud computing paradigm. Load balancers send service requests to functionally identical servers. Capacity and availability are increased by adding more servers and load balancing between them.

In this reference architecture two HAProxy nodes load balance the OpenStack services. HAproxy is an open source and free application that offers high availability and load balancing for TCP and HTTP based applications. The HAproxy nodes accept service requests on their management network interfaces via service-specific virtual IP addresses. The HAproxy nodes load balance the connection requests via their private interfaces to the controller nodes service interfaces. This is depicted in **Figure 4.2.1 Importance of the Load Balancer**. The version used in this reference architecture is supported and provided with a RHEL subscription.

This graphic illustrates the importance of the load balancers to the HA RHEL OSP 4 deployment. The load balancers field all client requests. There is no direct communication between the clients and the services. The load balancers forward the responses from the service listeners back to the clients. This ensures that client requests are never sent to down

or non-responsive service nodes. Furthermore, decoupling the services from the client requests allows the complete solution to scale on a per-service basis while maintaining centralized management and operation.

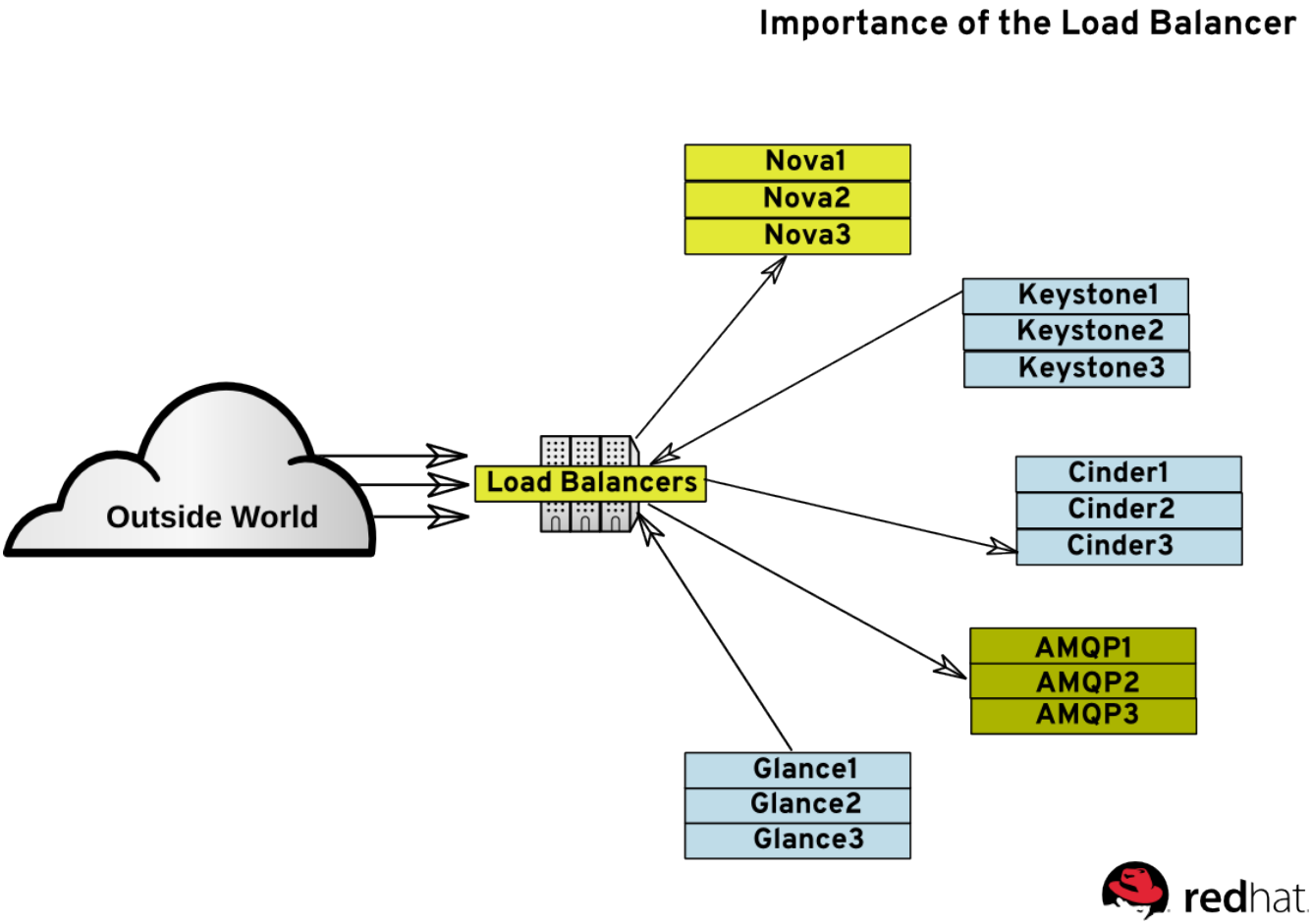

Figure 4.2.1: Importance of the Load Balancer

The Pacemaker resource manager monitors the load balancers for hardware or software failure. If Pacemaker detects a fault it moves the virtual IP addresses to the healthy node and attempts to restart the problem node. Pacemaker provides a layer of fault tolerance to the load balancers.

Each OpenStack service is associated with a single virtual IP address rather than a single failover IP address shared by all services. The advantage of this approach is that each load balancer can own half of the VIPs. This ensures that client connections are balanced across the load balancers in addition to being balanced across the controllers.

NOTE: An existing load balancing application or appliance can be used in place of the HAproxy nodes. HAproxy is shown in this reference architecture because it ships with Red Hat Enterprise Linux OpenStack Platform 4. Additionally, HAProxy is a light weight service. It can run directly on the OpenStack controller nodes alongside the OpenStack services. Neither of these options are described in this reference architecture.

This section describes the steps to configure the HAProxy nodes.

1. Add all OpenStack service ports to */etc/sysconfig/iptables* on both load balancers.

```
[root@rhos0 ~]# iptables -F

[root@rhos0 ~]# iptables -A INPUT -p tcp -m state --state NEW -m tcp
--dport 22 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 443 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 2049 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 3306 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 5404 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p udp -m udp --dport 5404 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 5405 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p udp -m udp --dport 5405 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 5000 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 5672 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 6080 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 6081 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8000 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8003 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8004 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8080 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8774 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8775 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8776 -j ACCEPT
```

```
[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 8777 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 9191 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 9292 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 9696 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 11211 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 27017 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p tcp -m tcp --dport 35357 -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -m state --state RELATED,ESTABLISHED -j
ACCEPT

[root@rhos0 ~]# iptables -A INPUT -p icmp -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -i lo -j ACCEPT

[root@rhos0 ~]# iptables -A INPUT -j REJECT --reject-with icmp-host-
prohibited

[root@rhos0 ~]# iptables -A FORWARD -j REJECT --reject-with icmp-host-
prohibited

[root@rhos0 ~]# service iptables save
```

2. Restart **`iptables`**.

```
[root@rhos0 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]
```

3. View open ports to verify firewall configuration.

```
[root@rhos0 ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW
tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:80
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:443
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:2049
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:3306
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5404
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:5404
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5405
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:5405
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5000
```

```
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5672
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:6080
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:6081
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8000
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8003
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8004
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8774
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8775
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8776
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8777
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:9191
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:9292
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:9696
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp
dpt:11211
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp
dpt:27017
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp
dpt:35357
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state
RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with
icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with
icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

4. Add the Server-LB and Server-HA RHN channels.

```
[root@rhos0 ~]# rhn-channel --user admin --password examplepasswd -a \
-c rhel-x86_64-server-ha-6 -c rhel-x86_64-server-lb-6
```

5. Install HAproxy and Pacemaker.

```
[root@rhos0 ~]# yum install -y pacemaker fence-agents resource-agents pcs
cman ccs haproxy
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
Stopping kdump:[  OK  ]
Starting kdump:[  OK  ]
```

6. Enable Pacemaker.

```
[root@rhos0 ~]# chkconfig pacemaker on
```

7. Allow local processes to bind to non-local IP addresses.

```
[root@rhos0 ~]# echo net.ipv4.ip_nonlocal_bind=1 >> /etc/sysctl.conf

[root@rhos0 ~]# sysctl -p
net.ipv4.ip_forward = 0
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.default.accept_source_route = 0
kernel.sysrq = 0
kernel.core_uses_pid = 1
net.ipv4.tcp_syncookies = 1
kernel.msgmnb = 65536
kernel.msgmax = 65536
kernel.shmmax = 68719476736
kernel.shmall = 4294967296
net.ipv4.ip_nonlocal_bind = 1

[root@rhos0 ~]# sysctl -a | grep ip_nonlocal
net.ipv4.ip_nonlocal_bind = 1
```

8. Populate */etc/haproxy/haproxy.cfg*.

   The HAproxy configuration file maps a front end virtual IP address to a backend pool of IP address and port combination and a balancing policy per service.

   NOTE: The default settings apply to all services except where otherwise noted. The mysql and qpid services have longer timeout values. The Qpid service backend uses a stick table. All Qpid connections through the load balancer stick to a single active server. This ensures proper Qpid functionality while keeping the messaging queue highly available.

```
global
  daemon
defaults
  maxconn  10000
  mode  tcp
  timeout  connect 2s
  timeout  client 10s
  timeout  server 10s
frontend vip-mysql
    bind 10.19.137.1:3306
    timeout client 90s
    default_backend mysql-vms
backend mysql-vms
    balance roundrobin
    timeout server 90s
    server rhos2-priv 172.16.2.102:3306 check inter 1s
    server rhos3-priv 172.16.2.103:3306 check inter 1s
    server rhos4-priv 172.16.2.104:3306 check inter 1s
frontend vip-qpid
    bind 10.19.137.2:5672
    timeout client 120s
    default_backend qpid-vms
```

```
backend qpid-vms
    stick on dst
    stick-table type ip size 2
    timeout server 120s
    server rhos2-priv 172.16.2.102:5672 check inter 1s
    server rhos3-priv 172.16.2.103:5672 check inter 1s
    server rhos4-priv 172.16.2.104:5672 check inter 1s
frontend vip-keystone-admin
    bind 10.19.137.3:35357
    default_backend keystone-admin-vms
backend keystone-admin-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:35357 check inter 1s
    server rhos3-priv 172.16.2.103:35357 check inter 1s
    server rhos4-priv 172.16.2.104:35357 check inter 1s
frontend vip-keystone-public
    bind 10.19.137.3:5000
    default_backend keystone-public-vms
backend keystone-public-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:5000 check inter 1
    server rhos3-priv 172.16.2.103:5000 check inter 1
    server rhos4-priv 172.16.2.104:5000 check inter 1
frontend vip-glance-api
    bind 10.19.137.5:9191
    default_backend glance-api-vms
backend glance-api-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:9191 check inter 1s
    server rhos3-priv 172.16.2.103:9191 check inter 1s
    server rhos4-priv 172.16.2.104:9191 check inter 1s
frontend vip-glance-registry
    bind 10.19.137.5:9292
    default_backend glance-registry-vms
backend glance-registry-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:9292 check inter 1s
    server rhos3-priv 172.16.2.103:9292 check inter 1s
    server rhos3-priv 172.16.2.104:9292 check inter 1s
frontend vip-cinder
    bind 10.19.137.6:8776
    default_backend cinder-vms
backend cinder-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:8776 check inter 1
    server rhos3-priv 172.16.2.103:8776 check inter 1
    server rhos4-priv 172.16.2.104:8776 check inter 1
frontend vip-neutron
    bind 10.19.137.8:9696
    default_backend neutron-vms
backend neutron-vms
    balance roundrobin
    server rhos4-priv 172.16.2.102:9696 check inter 1s
    server rhos4-priv 172.16.2.103:9696 check inter 1s
    server rhos5-priv 172.16.2.104:9696 check inter 1s
```

```
frontend vip-nova-vnc-novncproxy
    bind 10.19.137.9:6080
    default_backend nova-vnc-novncproxy-vms
backend nova-vnc-novncproxy-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:6080 check inter 1s
    server rhos3-priv 172.16.2.103:6080 check inter 1s
    server rhos4-priv 172.16.2.104:6080 check inter 1s
frontend vip-nova-metadata
    bind 10.19.137.9:8775
    default_backend nova-metadata-vms
backend nova-metadata-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:8775 check inter 1s
    server rhos3-priv 172.16.2.103:8775 check inter 1s
    server rhos4-priv 172.16.2.104:8775 check inter 1s
frontend vip-nova-api
    bind 10.19.137.9:8774
    default_backend nova-api-vms
backend nova-api-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:8774 check inter 1s
    server rhos3-priv 172.16.2.103:8774 check inter 1s
    server rhos4-priv 172.16.2.104:8774 check inter 1s
frontend vip-horizon
    bind 10.19.137.10:80
    default_backend horizon-vms
backend horizon-vms
    balance roundrobin
    server rhos2-priv 172.16.2.102:80 check inter 1s
    server rhos3-priv 172.16.2.103:80 check inter 1s
    server rhos4-priv 172.16.2.104:80 check inter 1s
```

9. Create and start the cluster on both load balancers.

> **NOTE:** These commands should be executed at the same time on both load balancers.

```
[root@rhos0 ~]# pcs cluster setup --name rhos4-lb rhos0-priv rhos1-priv

[root@rhos0 ~]# pcs cluster start
Starting Cluster...
```

10. Do not proceed until both cluster nodes are online according to the **`pcs status`** command.

```
[root@rhos0 ~]# pcs cluster status
Cluster name: rhos4-lb
Last updated: Thu Jun 12 10:50:15 2014
Last change: Thu Jun 12 10:50:14 2014 via cibadmin on rhos1-priv
Stack: cman
Current DC: rhos1-priv - partition with quorum
```

```
Version: 1.1.10-14.el6_5.3-368c726
2 Nodes configured

Online: [ rhos0-priv rhos1-priv ]
```

11. Add the fence devices.

NOTE: Perform steps 11-17 on only one node.

```
[root@rhos0 ~]# pcs stonith create rhos0-fence fence_ipmilan params \
login="root" passwd="examplepasswd" action="reboot" \
ipaddr="10.19.143.162" lanplus="" verbose="" pcmk_host_list=rhos0-priv

[root@rhos0 ~]# pcs stonith create rhos1-fence fence_ipmilan params \
login="root" passwd="examplepasswd" action="reboot" \
ipaddr="10.19.143.163" lanplus="" verbose="" pcmk_host_list=rhos1-priv

[root@rhos0 ~]# pcs stonith show
rhos0-fence(stonith:fence_ipmilan):    Started
rhos1-fence(stonith:fence_ipmilan):    Started
```

NOTE: This reference architecture uses Dell Remote Access Controller cards as IPMI fence devices. Use an fence agent appropriate to the environment. **Chapter 4 of the RHEL HA Add-on Guide** lists available fencing agents and provides configuration details.

12. Cluster the HAProxy service.

```
[root@rhos0 ~]# pcs resource create lb-haproxy lsb:haproxy --clone
```

13. Do not proceed until the clone set is started on both nodes.

```
[root@rhos0 ~]# pcs resource show
Clone Set: lb-haproxy-clone [lb-haproxy]
Started: [ rhos0-priv rhos1-priv ]
```

14. Create the virtual IP addresses on an active node.

```
[root@rhos0 ~]# pcs resource create vip-mysql IPaddr2 ip=10.19.137.1

[root@rhos0 ~]# pcs resource create vip-qpid IPaddr2 ip=10.19.137.2

[root@rhos0 ~]# pcs resource create vip-keystone IPaddr2 ip=10.19.137.3

[root@rhos0 ~]# pcs resource create vip-glance IPaddr2 ip=10.19.137.5

[root@rhos0 ~]# pcs resource create vip-cinder IPaddr2 ip=10.19.137.6

[root@rhos0 ~]# pcs resource create vip-neutron IPaddr2 ip=10.19.137.8
```

```
[root@rhos0 ~]# pcs resource create vip-nova IPaddr2 ip=10.19.137.9

[root@rhos0 ~]# pcs resource create vip-horizon IPaddr2 ip=10.19.137.10
```

15. Verify all resources are started and where they are running.

```
[root@rhos0 ~]# pcs status
Cluster name: rhos4-lb
Last updated: Thu Jun 12 11:07:37 2014
Last change: Thu Jun 12 10:55:14 2014 via cibadmin on rhos1-priv
Stack: cman
Current DC: rhos1-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
2 Nodes configured
12 Resources configured

Online: [ rhos0-priv rhos1-priv ]

Full list of resources:

 rhos0-fence     (stonith:fence_ipmilan):    Started rhos0-priv
 rhos1-fence     (stonith:fence_ipmilan):    Started rhos1-priv
 vip-mysql (ocf::heartbeat:IPaddr2):   Started rhos0-priv
 vip-qpid  (ocf::heartbeat:IPaddr2):   Started rhos1-priv
 vip-keystone    (ocf::heartbeat:IPaddr2):   Started rhos0-priv
 vip-glance(ocf::heartbeat:IPaddr2):   Started rhos1-priv
 vip-cinder(ocf::heartbeat:IPaddr2):   Started rhos0-priv
 vip-neutron     (ocf::heartbeat:IPaddr2):   Started rhos1-priv
 vip-nova  (ocf::heartbeat:IPaddr2):   Started rhos0-priv
 vip-horizon     (ocf::heartbeat:IPaddr2):   Started rhos1-priv
 Clone Set: lb-haproxy-clone [lb-haproxy]
     Started: [ rhos0-priv rhos1-priv ]
```

16. Ping the virtual IPs to verify they are online.

```
[root@rhos0 ~]# for i in mysql qpid keystone glance cinder neutron nova
horizon ; do  ping -c 3 rhos-$i-vip; done

PING rhos-mysql-vip.cloud.lab.eng.bos.redhat.com (10.19.137.1) 56(84)
bytes of data.
64 bytes from vip-mysql (10.19.137.1): icmp_seq=1 ttl=64 time=0.040 ms
64 bytes from vip-mysql (10.19.137.1): icmp_seq=2 ttl=64 time=0.031 ms
64 bytes from vip-mysql (10.19.137.1): icmp_seq=3 ttl=64 time=0.030 ms

--- rhos-mysql-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.030/0.033/0.040/0.008 ms
PING rhos-qpid-vip.cloud.lab.eng.bos.redhat.com (10.19.137.2) 56(84)
bytes of data.
64 bytes from vip-qpid (10.19.137.2): icmp_seq=1 ttl=64 time=0.451 ms
64 bytes from vip-qpid (10.19.137.2): icmp_seq=2 ttl=64 time=0.226 ms
64 bytes from vip-qpid (10.19.137.2): icmp_seq=3 ttl=64 time=0.212 ms
```

```
--- rhos-qpid-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.212/0.296/0.451/0.110 ms
PING rhos-keystone-vip.cloud.lab.eng.bos.redhat.com (10.19.137.3) 56(84)
bytes of data.
64 bytes from vip-keystone (10.19.137.3): icmp_seq=1 ttl=64 time=0.030 ms
64 bytes from vip-keystone (10.19.137.3): icmp_seq=2 ttl=64 time=0.031 ms
64 bytes from vip-keystone (10.19.137.3): icmp_seq=3 ttl=64 time=0.030 ms

--- rhos-keystone-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.030/0.030/0.031/0.004 ms
PING rhos-glance-vip.cloud.lab.eng.bos.redhat.com (10.19.137.5) 56(84)
bytes of data.
64 bytes from vip-glance (10.19.137.5): icmp_seq=1 ttl=64 time=1.21 ms
64 bytes from vip-glance (10.19.137.5): icmp_seq=2 ttl=64 time=0.223 ms
64 bytes from vip-glance (10.19.137.5): icmp_seq=3 ttl=64 time=0.224 ms

--- rhos-glance-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.223/0.552/1.210/0.465 ms
PING rhos-cinder-vip.cloud.lab.eng.bos.redhat.com (10.19.137.6) 56(84)
bytes of data.
64 bytes from vip-cinder (10.19.137.6): icmp_seq=1 ttl=64 time=0.030 ms
64 bytes from vip-cinder (10.19.137.6): icmp_seq=2 ttl=64 time=0.031 ms
64 bytes from vip-cinder (10.19.137.6): icmp_seq=3 ttl=64 time=0.030 ms

--- rhos-cinder-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.030/0.030/0.031/0.004 ms
PING rhos-neutron-vip.cloud.lab.eng.bos.redhat.com (10.19.137.8) 56(84)
bytes of data.
64 bytes from vip-neutron (10.19.137.8): icmp_seq=1 ttl=64 time=1.38 ms
64 bytes from vip-neutron (10.19.137.8): icmp_seq=2 ttl=64 time=0.218 ms
64 bytes from vip-neutron (10.19.137.8): icmp_seq=3 ttl=64 time=0.225 ms

--- rhos-neutron-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.218/0.609/1.384/0.548 ms
PING rhos-nova-vip.cloud.lab.eng.bos.redhat.com (10.19.137.9) 56(84)
bytes of data.
64 bytes from vip-nova (10.19.137.9): icmp_seq=1 ttl=64 time=0.030 ms
64 bytes from vip-nova (10.19.137.9): icmp_seq=2 ttl=64 time=0.040 ms
64 bytes from vip-nova (10.19.137.9): icmp_seq=3 ttl=64 time=0.030 ms

--- rhos-nova-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.030/0.033/0.040/0.006 ms
PING rhos-horizon-vip.cloud.lab.eng.bos.redhat.com (10.19.137.10) 56(84)
bytes of data.
64 bytes from vip-horizon (10.19.137.10): icmp_seq=1 ttl=64 time=2.18 ms
64 bytes from vip-horizon (10.19.137.10): icmp_seq=2 ttl=64 time=0.232 ms
64 bytes from vip-horizon (10.19.137.10): icmp_seq=3 ttl=64 time=0.236 ms

--- rhos-horizon-vip.cloud.lab.eng.bos.redhat.com ping statistics ---
```

```
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.232/0.884/2.184/0.919 ms
```

17. Display the cluster resources.

```
[root@rhos0 ~]# pcs resource show --full
 Resource: vip-mysql (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.1
  Operations: monitor interval=60s (vip-mysql-monitor-interval-60s)
 Resource: vip-qpid (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.2
  Operations: monitor interval=60s (vip-qpid-monitor-interval-60s)
 Resource: vip-keystone (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.3
  Operations: monitor interval=60s (vip-keystone-monitor-interval-60s)
 Resource: vip-glance (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.5
  Operations: monitor interval=60s (vip-glance-monitor-interval-60s)
 Resource: vip-cinder (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.6
  Operations: monitor interval=60s (vip-cinder-monitor-interval-60s)
 Resource: vip-neutron (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.8
  Operations: monitor interval=60s (vip-neutron-monitor-interval-60s)
 Resource: vip-nova (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.9
  Operations: monitor interval=60s (vip-nova-monitor-interval-60s)
 Resource: vip-horizon (class=ocf provider=heartbeat type=IPaddr2)
  Attributes: ip=10.19.137.10
  Operations: monitor interval=60s (vip-horizon-monitor-interval-60s)
```

4.3 Configure the Controller Nodes

In this reference architecture the OpenStack controller nodes run the following services:

- The MySQL state database
- The Qpid message broker
- The Memcached object cache
- The Neutron API server and networking agents
- The Horizon web interface
- API and scheduler services including Cinder, Glance, and Keystone

Three servers participate in the controller node cluster. Most of the highly available services run as active/active clone sets. Connections are load balanced between them by the load balancers. Other services run in an active/passive arrangement. The details for each service are described in section **2.2.2.4 OpenStack Services**.

4.3.1 Configure the Network Bridges

The Open vSwitch plugin defines network bridges and ports for virtual machines and their

associated services. This reference architecture requires two bridges. The integration bridge – **br-int** – connects to the virtual machines internally. The external bridge – **br-p3p1** – connects the virtual machines with the external network and allows tenant traffic between hypervisors. The bridges are defined in the first step so that highly available OpenStack services are not interrupted later in the process.

1. Install Neutron and the Open vSwitch plugin.

```
[root@rhos2 ~]# yum install -y openstack-neutron openstack-neutron-
openvswitch
```

2. Create initialization scripts for the bridge interfaces.

> **NOTE:** In this example the **p3p1** interface is added to the **br-p3p1** bridge. The **br-p3p1** bridge does not need an IP address. However, all **p3p1** interfaces are added to VLANs 1000-1010 on the lab switches. All interfaces must be in UP state with an active link. The interfaces' tagged VLAN traffic for VLANs 1000-1010 were enabled on the switch ports.

```
[root@rhos2 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-br-p3p1 << EOF
> DEVICE=br-p3p1
> DEVICETYPE=ovs
> TYPE=OVSBridge
> ONBOOT=yes
> OVSBOOTPROTO=static
> EOF

[root@rhos2 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-p3p1 << EOF
> DEVICE=p3p1
> NM_CONTROLLED=no
> ONBOOT=yes
> DEVICETYPE=ovs
> OVS_BRIDGE=br-p3p1
> TYPE=OVSPort
> EOF

[root@rhos2 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-br-int << EOF
> DEVICE=br-int
> DEVICETYPE=ovs
> TYPE=OVSBridge
> ONBOOT=yes
> BOOTPROTO=none
> EOF
```

3. Restart networking to create the bridges.

```
[root@rhos2 ~]# service network restart
Shutting down interface em1:                               [  OK  ]
Shutting down interface em2:                               [  OK  ]
Shutting down interface p3p1:  /etc/openvswitch/conf.db does not
```

```
exist ... (warning).
Creating empty database /etc/openvswitch/conf.db           [  OK  ]
Starting ovsdb-server                                      [  OK  ]
Configuring Open vSwitch system IDs                        [  OK  ]
Inserting openvswitch module                               [  OK  ]
Starting ovs-vswitchd                                      [  OK  ]
Enabling remote OVSDB managers                             [  OK  ]
                                                           [  OK  ]
Shutting down interface p3p2:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface br-int:                              [  OK  ]
Bringing up interface br-p3p1:                             [  OK  ]
Bringing up interface em1:
Determining IP information for em1... done.
                                                           [  OK  ]
Bringing up interface em2:  Determining if ip address 172.16.2.104 is
already in use for device em2...
                                                           [  OK  ]
Bringing up interface p3p1:  RTNETLINK answers: File exists
                                                           [  OK  ]
Bringing up interface p3p2:  Determining if ip address 172.31.139.104 is
already in use for device p3p2...
                                                           [  OK  ]
```

4. Verify bridge creation.

```
[root@rhos2 ~]# ovs-vsctl show
9355e380-8e59-4b3d-8a0d-9046ea15a28f
    Bridge br-int
        Port br-int
            Interface br-int
                type: internal
    Bridge "br-p3p1"
        Port "br-p3p1"
            Interface "br-p3p1"
                type: internal
        Port "p3p1"
            Interface "p3p1"
    ovs_version: "1.11.0"

[root@rhos2 ~]# ip addr show dev br-p3p1
8: br-p3p1: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN
    link/ether e0:db:55:74:41:33 brd ff:ff:ff:ff:ff:ff
    inet6 2620:52:0:1388:e2db:55ff:fe74:4133/64 scope global tentative
dynamic
       vlid_lft 2591999sec preferred_lft 604799sec
    inet6 fe80::a85e:4fff:fee7:6925/64 scope link
       valid_lft forever preferred_lft forever

[root@rhos2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-br-p3p1
DEVICE=br-p3p1
DEVICETYPE=ovs
TYPE=OVSBridge
```

```
ONBOOT=yes
OVSBOOTPROTO=static
```

4.3.2 Prepare the Controller Nodes for HA

In this reference architecture the controller nodes form a high availability cluster to manage the MySQL state database and Qpid messaging. Prepare the controller nodes to participate in the HA cluster by adding firewall rules and connecting them to shared storage.

1. Open Qpid and MySQL firewall ports, save changes, and restart **`iptables`**.

```
[root@rhos2 ~]# iptables -I INPUT -p udp -m udp --dport 5405 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 5405 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 3306 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 5672 -j ACCEPT

[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]
```

2. Add the messaging, high availability, and load balancing channels.

```
[root@rhos2 ~]# rhn-channel --user admin --password examplepasswd -a \
-c rhel-x86_64-server-ha-6 -c rhel-x86_64-server-6-mrg-messaging-2 \
-c rhel-x86_64-server-lb-6

[root@rhos2 ~]# rhn-channel -l
rhel-x86_64-server-6
rhel-x86_64-server-6-mrg-messaging-2
rhel-x86_64-server-6-ost-4
rhel-x86_64-server-ha-6
rhel-x86_64-server-lb-6
```

3. Verify that all packages are up to date.

   NOTE: The **`iproute`** package must be at version 2.6.32-130 or higher. The package name should include **`netns`**.

```
[root@rhos7 ~]# yum update -y
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
```

```
[root@rhos7 ~]# rpm -q iproute
iproute-2.6.32-130.el6ost.netns.3.x86_64
```

4. Install clustering, messaging, database, and storage software.

```
[root@rhos2 ~]# yum install -y pacemaker fence-agents resource-agents \
pcs cman ccs mysql mysql-server iscsi-initiator-utils qpid-tools \
qpid-cpp-server qpid-cpp-client-devel

Stopping kdump:[  OK  ]
Starting kdump:[  OK  ]
```

5. Connect the controller nodes to shared storage. The steps required to complete this task vary by shared storage type.

> **NOTE:** In this reference architecture the controller nodes attached to a 500 GB iSCSI target. **Appendix C: Configure Shared Storage** describes these steps for the client. Server configuration is hardware specific.

At the completion of this step all controller nodes should see the same inactive logical volume and volume group in **vgs** output, and their *volume_list* entry in */etc/lvm/lvm.conf* should not activate it at boot. In this reference architecture the logical volume is named *mysql-vol/mysql-lv*.

```
[root@rhos2 ~]# lvs
LV       VG   Attr LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
mysql-lv mysql-vol -wi------- 200.00g
rootvol  myvg      -wi-ao---- 134.94g
```

6. Modify the volume_list entry in */etc/lvm/lvm.conf* so that only local volumes groups are activated automatically. This ensures multiple cluster nodes do not activate the database logical volume at the same time.

```
[root@rhos2 ~]# grep ^volume_list /etc/lvm/lvm.conf
volume_list = [ "myvg", "@rhos2" ]
```

> **NOTE:** *myvg* is the default name of the local volume group. Include it in *volume_list*.

7. Recreate the initial RAM disk to ensure the volume group is not activated at boot.

```
[root@rhos2 ~]# cp /boot/initramfs-$(uname -r).img /boot/initramfs-$
(uname -r).img.orig

[root@rhos2 ~]# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

8. Configure */etc/qpidd.conf* on all controller nodes.

```
port=5672
max-connections=65535
worker-threads=17
connection-backlog=65535
auth=no
realm=QPID
```

NOTE: This reference architecture did not require Qpid authentication. Refer to the **Red Hat Enterprise Linux OpenStack Platform 4 Installation Guide** for instructions on configuring SASL Plain authentication for Qpid.

9. Enable, create, and start the cluster. Execute this command on all controller nodes at the same time.

```
[root@rhos2 ~]# chkconfig pacemaker on

[root@rhos2 ~]# pcs cluster setup --name osp4-controller rhos2-priv \
rhos3-priv rhos4-priv

[root@rhos2 ~]# pcs cluster start
Starting Cluster...
```

10. Verify that all controller nodes are online before proceeding.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
WARNING: no stonith devices and stonith-enabled is not false
Last updated: Thu Jun 12 11:19:23 2014
Last change: Thu Jun 12 11:19:07 2014 via crmd on rhos3-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
0 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:
```

11. On one controller node, configure all of the fence devices.

NOTE: Commands 11-19 should be executed on a single controller node.

```
[root@rhos2 ~]# pcs stonith create rhos2-fence fence_ipmilan params \
login="root" passwd="examplepasswd" action="reboot" \
ipaddr="10.19.143.167" lanplus="" verbose="" pcmk_host_list=rhos2-priv

[root@rhos2 ~]# pcs stonith create rhos3-fence fence_ipmilan params \
login="root" passwd="examplepasswd" action="reboot" \
ipaddr="10.19.143.170" lanplus="" verbose="" pcmk_host_list=rhos3-priv
```

```
[root@rhos2 ~]# pcs stonith create rhos4-fence fence_ipmilan params  \
login="root" passwd="examplepasswd" action="reboot" \
ipaddr="10.19.143.171" lanplus="" verbose="" pcmk_host_list=rhos4-priv

[root@rhos2 ~]# pcs stonith show
 rhos2-fence     (stonith:fence_ipmilan):    Started
 rhos3-fence     (stonith:fence_ipmilan):    Started
 rhos4-fence     (stonith:fence_ipmilan):    Started
```

12. On one controller, create the active/active Qpid cluster resource. As previously mentioned, the Qpid cluster resource is active/active but acts as a hot standby resource due to the HAproxy stick table setting.

```
[root@rhos2 ~]# pcs resource create qpidd lsb:qpidd --clone

[root@rhos2 ~]# pcs resource show
Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

13. On one controller node, create an exclusive *mysql-ha-lvm* resource for the shared volume group. Do not proceed to the next step until the resource group is started.

```
[root@rhos2 ~]# pcs resource create mysql-ha-lvm ocf:heartbeat:LVM
volgrpname=mysql-vol exclusive=true --group mysql-group

[root@rhos2 ~]# pcs resource show
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started
```

14. Identify which controller node currently owns the MySQL resource group.

```
[root@rhos2 ~]# pcs status | grep mysql-ha-lvm
  mysql-ha-lvm   (ocf::heartbeat:LVM):  Started rhos2-priv
```

15. Add a MySQL filesystem resource from the controller node identified in the previous step. This resource mounts the clustered logical volume to */var/lib/mysql*.

```
[root@rhos2 ~]# pcs resource create mysql-fs Filesystem \
device="/dev/mysql-vol/mysql-lv" directory="/var/lib/mysql" \
fstype="ext4" --group mysql-group

[root@rhos2 ~]# pcs resource show
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started
     mysql-fs    (ocf::heartbeat:Filesystem):Started

[root@rhos2 ~]# mount | grep mysql
/dev/mapper/mysql--vol-mysql--lv on /var/lib/mysql type ext4 (rw)
```

16. Set the SELinux context on */var/lib/mysql*.

```
[root@rhos2 ~]# restorecon -Rv /var/lib
restorecon reset /var/lib/mysql context system_u:object_r:file_t:s0-
>system_u:object_r:mysqld_db_t:s0
restorecon reset /var/lib/mysql/lost+found context
system_u:object_r:file_t:s0->system_u:object_r:mysqld_db_t:s0
```

17. Copy the MySQL configuration file to the clustered filesystem and increase the connection allowance to accommodate a cloud infrastructure.

```
[root@rhos2 ~]# cp /etc/my.cnf /var/lib/mysql/

[root@rhos2 ~]# openstack-config --set /var/lib/mysql/my.cnf mysqld
max_connections 10000
```

18. Add a clustered MySQL service resource. Specify the configuration file on the shared filesystem. Do not proceed until the service starts.

```
[root@rhos2 ~]# pcs resource create mysql-db mysql
config=/var/lib/mysql/my.cnf enable_creation=1 op start timeout=120s
--group mysql-group

[root@rhos2 ~]# pcs status | grep mysql-db | grep Started
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
```

19. Set the MySQL password and access permissions from the service-owning controller node.

```
[root@rhos2 ~]# mysqladmin -u root password redhat

[root@rhos2 ~]# mysql -u root --password=redhat -e "GRANT ALL PRIVILEGES
ON *.* TO 'root'@'%' IDENTIFIED by 'redhat' WITH GRANT OPTION;"

[root@rhos2 ~]# mysql -u root --password=redhat -e "drop user ''@'$
(hostname)';"

[root@rhos2 ~]# mysql -u root --password=redhat -e "drop user 'root'@'$
(hostname)';"

[root@rhos2 ~]# mysql -u root --password=redhat -e "drop user
''@'localhost';"

[root@rhos2 ~]# mysql -u root --password=redhat -e "drop user
'root'@'localhost';"

[root@rhos2 ~]# mysql -u root --password=redhat -e "drop user
'root'@'127.0.0.1';"

[root@rhos2 ~]# mysql -u root --password=redhat -e "flush privileges;"

[root@rhos2 ~]# mysql -u root --password=redhat mysql -e "select
```

```
User,Host,Password from user"
+------+------+-------------------------------------------+
| User | Host | Password                                  |
+------+------+-------------------------------------------+
| root | %    | *84BB5DF4823DA319BBF86C99624479A198E6EEE9 |
+------+------+-------------------------------------------+
```

20. Verify that the Qpid and MySQL resources are started.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 11:24:13 2014
Last change: Thu Jun 12 11:20:12 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
9 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
```

21. Verify that all three MySQL services are functional: the process is running, the file system is mounted, and the volume group is active.

```
[root@rhos2 ~]# ps -ef | grep mysql
root      6875     1  0 11:20 ?        00:00:00 /bin/sh
/usr/bin/mysqld_safe --defaults-file=/var/lib/mysql/my.cnf --pid-
file=/var/run/mysql/mysqld.pid --socket=/var/lib/mysql/mysql.sock
--datadir=/var/lib/mysql --log-error=/var/log/mysqld.log --user=mysql
mysql     6976  6875  0 11:20 ?        00:00:00 /usr/libexec/mysqld
--defaults-file=/var/lib/mysql/my.cnf --basedir=/usr
--datadir=/var/lib/mysql --user=mysql --log-error=/var/log/mysqld.log
--pid-file=/var/run/mysql/mysqld.pid --socket=/var/lib/mysql/mysql.sock
root      7554 31413  0 11:24 pts/1    00:00:00 grep mysql

[root@rhos2 ~]# mount | grep mysql
/dev/mapper/mysql--vol-mysql--lv on /var/lib/mysql type ext4 (rw)

[root@rhos2 ~]# vgs
  VG        #PV #LV #SN Attr   VSize   VFree
  mysql-vol   1   1   0 wz--n- 500.01g 300.01g
  myvg        1   1   0 wz--n- 134.94g      0
```

22. Verify that **qpidd** is running on all servers.

```
[root@rhos2 ~]# ps -ef | grep qpid
qpidd     6245     1 52 11:19 ?        00:00:34 /usr/sbin/qpidd --data-
dir /var/lib/qpidd --daemon
root      7085 31422  0 11:20 pts/1    00:00:00 grep qpid
```

23. Test Qpid functionality through the load balancers by running **qpid-perftest** against the Qpid virtual IP address.

> **NOTE: qpid-perftest** is installed by the **qpid-cpp-client-devel** package.

```
[root@rhos2 ~]# qpid-perftest -s -b rhos-qpid-vip -p 5672 --iterations 2
--tcp-nodelay
54189.4    53493.8    107957     105.426
55654.5    55635.3    111255     108.647
Averages:
54922 54564.5    109606     107.037
```

4.3.3 Implement HA Keystone

The Keystone service manages identities and controls access to resources in OpenStack. This section describes the steps to implement Keystone on the three controller nodes as an active/active HA cluster.

1. Add the Keystone ports to all three controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 5000  -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 35357 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep -E '5000|35357'
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp
dpt:35357
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:5000
```

3. Install Keystone on all three controller nodes.

```
[root@rhos2 ~]# yum install -y openstack-selinux openstack-keystone
```

4. Create the Keystone database and user on a single controller node.

```
[root@rhos2 ~]# mysql --user=root --password=redhat --host=rhos-mysql-vip
-e "CREATE DATABASE keystone;"

[root@rhos2 ~]# mysql --user=root --password=redhat --host=rhos-mysql-vip
-e "GRANT ALL ON keystone.* TO 'keystone'@'%' IDENTIFIED BY
'keystonetest';"

[root@rhos2 ~]# mysql --user=root --password=redhat --host=rhos-mysql-vip
-e "GRANT ALL ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY
'keystonetest';"

[root@rhos2 ~]# mysql --user=root --password=redhat --host=rhos-mysql-vip
-e "FLUSH PRIVILEGES;"
```

5. On a single controller node, define an authorization token to bootstrap Keystone.

```
[root@rhos2 ~]# export OS_SERVICE_TOKEN=$(openssl rand -hex 10)

[root@rhos2 ~]# echo $OS_SERVICE_TOKEN > /root/ks_admin_token

[root@rhos2 ~]# cat /root/ks_admin_token
de42bc247e42392b07fb
```

6. Copy the authorization token to the other controller nodes.

```
[root@rhos2 ~]# scp /root/ks_admin_token rhos3:/root/ks_admin_token
ks_admin_token 100%   21     0.0KB/s   00:00
[root@rhos2 ~]# scp /root/ks_admin_token rhos4:/root/ks_admin_token
ks_admin_token 100%   21     0.0KB/s   00:00
```

7. Export the token as an environment variable on ALL controller nodes. In this example, the environment variable is executed on *rhos3*.

```
[root@rhos3 ~]# export OS_SERVICE_TOKEN=$(cat /root/ks_admin_token)

[root@rhos3 ~]# echo $OS_SERVICE_TOKEN
de42bc247e42392b07fb
```

8. Add the Keystone service endpoints to the OpenStack configuration. Both the admin and public endpoints should use the Keystone virtual IP address. Execute these commands on ALL controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/keystone/keystone.conf
DEFAULT admin_token $OS_SERVICE_TOKEN

[root@rhos2 ~]# openstack-config --set /etc/keystone/keystone.conf sql
connection mysql://keystone:keystonetest@rhos-mysql-vip/keystone
```

```
[root@rhos2 ~]# openstack-config --set /etc/keystone/keystone.conf
DEFAULT admin_endpoint 'http://rhos-keystone-vip:%(admin_port)s/'

[root@rhos2 ~]# openstack-config --set /etc/keystone/keystone.conf
DEFAULT public_endpoint 'http://rhos-keystone-vip:%(public_port)s/'
```

9. Create an SSL key for the Keystone user on a single controller.

```
[root@rhos2 ~]# keystone-manage pki_setup --keystone-user keystone
--keystone-group keystone
2014-04-28 12:04:24.616 8967 INFO keystone.common.openssl [-] openssl
genrsa -out /etc/keystone/ssl/certs/cakey.pem 2048
Generating RSA private key, 2048 bit long moduluse is 65537 (0x10001)
2014-04-28 12:04:25.169 8967 INFO keystone.common.openssl [-] openssl req
-new -x509 -extensions v3_ca -key /etc/keystone/ssl/certs/cakey.pem
-out /etc/keystone/ssl/certs/ca.pem -days 3650 -config
/etc/keystone/ssl/certs/openssl.conf -subj
/C=US/ST=Unset/L=Unset/O=Unset/CN=www.example.com

2014-04-28 12:04:25.186 8967 INFO keystone.common.openssl [-] openssl
genrsa -out /etc/keystone/ssl/private/signing_key.pem 2048
Generating RSA private key, 2048 bit long moduluse is 65537 (0x10001)

2014-04-28 12:04:25.642 8967 INFO keystone.common.openssl [-] openssl req
-key /etc/keystone/ssl/private/signing_key.pem -new -out
/etc/keystone/ssl/certs/req.pem -config
/etc/keystone/ssl/certs/openssl.conf -subj
/C=US/ST=Unset/L=Unset/O=Unset/CN=www.example.com

2014-04-28 12:04:25.656 8967 INFO keystone.common.openssl [-] openssl ca
-batch -out /etc/keystone/ssl/certs/signing_cert.pem -config
/etc/keystone/ssl/certs/openssl.conf -days 3650d -cert
/etc/keystone/ssl/certs/ca.pem -keyfile /etc/keystone/ssl/certs/cakey.pem
-infiles /etc/keystone/ssl/certs/req.pem

Using configuration from /etc/keystone/ssl/certs/openssl.conf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'US'
stateOrProvinceName   :ASN.1 12:'Unset'
localityName          :ASN.1 12:'Unset'
organizationName      :ASN.1 12:'Unset'
commonName            :ASN.1 12:'www.example.com'
Certificate is to be certified until Apr 25 17:04:25 2024 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
```

10. Change SSL key ownership to the keystone user and sync the database.

```
[root@rhos2 ~]# chown -R keystone:keystone /var/log/keystone
/etc/keystone/ssl/
```

```
[root@rhos2 ~]# su keystone -s /bin/sh -c "keystone-manage db_sync"
```

11. Copy the SSL key to the other controllers.

```
[root@rhos2 ~]# rsync -zvr /etc/keystone/ssl rhos3:/etc/keystone
sending incremental file list
ssl/
ssl/certs/
ssl/certs/01.pem
ssl/certs/ca.pem
ssl/certs/cakey.pem
ssl/certs/index.txt
ssl/certs/index.txt.attr
ssl/certs/index.txt.old
ssl/certs/openssl.conf
ssl/certs/req.pem
ssl/certs/serial
ssl/certs/serial.old
ssl/certs/signing_cert.pem
ssl/private/
ssl/private/signing_key.pem

sent 10498 bytes  received 252 bytes  1954.55 bytes/sec
total size is 16185  speedup is 1.51

[root@rhos2 ~]# rsync -zvr /etc/keystone/ssl rhos4:/etc/keystone
sending incremental file list
ssl/
ssl/certs/
ssl/certs/01.pem
ssl/certs/ca.pem
ssl/certs/cakey.pem
ssl/certs/index.txt
ssl/certs/index.txt.attr
ssl/certs/index.txt.old
ssl/certs/openssl.conf
ssl/certs/req.pem
ssl/certs/serial
ssl/certs/serial.old
ssl/certs/signing_cert.pem
ssl/private/
ssl/private/signing_key.pem

sent 10498 bytes  received 252 bytes  3071.43 bytes/sec
total size is 16185  speedup is 1.51
```

12. Ensure the Keystone directories are owned by the *keystone* user. Execute this command on ALL controller nodes.

```
[root@rhos3 ~]# chown -R keystone:keystone /var/log/keystone
/etc/keystone/ssl/
```

13. On a single controller create the Keystone cluster resource.

```
[root@rhos2 ~]# pcs resource create keystone lsb:openstack-keystone
--clone
```

14. Verify that the Keystone cluster resource starts on all controller nodes before proceeding.

```
[root@rhos2 ~]# pcs status | grep -A 2 keystone-clone | grep Started |
grep rhos2 | grep rhos3 | grep rhos4

Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

15. Create a user environment configuration file for the OpenStack administrator named */root/keystonerc_admin*. Create this file on ALL controller nodes.

```
export OS_USERNAME=admin
export OS_TENANT_NAME=admin
export OS_PASSWORD=keystonetest
export OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
export PS1='[\u@\h \W(keystone_admin)]\$ '
```

16. Prepare to create Keystone resources by exporting the endpoint and token.

NOTE: Execute commands 16-21 on a single controller node.

```
[root@rhos2 ~]# export OS_SERVICE_ENDPOINT="http://rhos-keystone-
vip:35357/v2.0"

[root@rhos2 ~]# echo -e "OS service endpoint: $OS_SERVICE_ENDPOINT"
OS service endpoint: http://rhos-keystone-vip:35357/v2.0

[root@rhos2 ~]# export OS_SERVICE_TOKEN=$(cat /root/ks_admin_token)

[root@rhos2 ~]# echo -e "OS service token: $OS_SERVICE_TOKEN"
OS service token: de42bc247e42392b07fb
```

17. Create the admin and services tenants.

```
[root@rhos2 ~]# keystone tenant-create --name admin
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|   enabled   |               True               |
|      id     | 6824420035fd463da599bc91fd5d213a |
|     name    |              admin               |
+-------------+----------------------------------+

[root@rhos2 ~]# keystone tenant-create --name services --description
"Services Tenant"
```

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |         Services Tenant          |
|   enabled   |               True               |
|      id     | 7ebb91e93b2c49118537f9ef999fe941 |
|     name    |             services             |
+-------------+----------------------------------+

[root@rhos2 ~]# keystone tenant-list
+----------------------------------+----------+---------+
|                id                |   name   | enabled |
+----------------------------------+----------+---------+
| 6824420035fd463da599bc91fd5d213a |  admin   |   True  |
| 7ebb91e93b2c49118537f9ef999fe941 | services |   True  |
+----------------------------------+----------+---------+
```

18. Create and view the *admin* role.

```
[root@rhos2 ~]# keystone role-create --name admin
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|    id    | 3f07994ed38e4c6cb32f349827d90f7c |
|   name   |              admin               |
+----------+----------------------------------+

[root@rhos2 ~]# keystone role-list
+----------------------------------+----------+
|                id                |   name   |
+----------------------------------+----------+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
| da4df019550b457ab3d90c8fe7d25dd5 |  admin   |
+----------------------------------+----------+
```

19. Create the *admin* user.

```
[root@rhos2 ~]# keystone user-create --name admin --pass keystonetest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 97999e2533214d559cf09c94f86bfd75 |
|   name   |              admin               |
+----------+----------------------------------+

[root@rhos2 ~]# keystone user-list
+----------------------------------+-------+---------+-------+
|                id                |  name | enabled | email |
+----------------------------------+-------+---------+-------+
| 97999e2533214d559cf09c94f86bfd75 | admin |   True  |       |
+----------------------------------+-------+---------+-------+
```

20. Associate the *admin* role, tenant, and user.

```
[root@rhos2 ~]# keystone user-role-add --user admin --role admin --tenant
admin

[root@rhos2 ~]# keystone user-role-list --user admin --tenant admin
+----------------------------------+-------+----------------------------+
|                id                |  name |             user_id
|           tenant_id              |
+----------------------------------+-------+----------------------------+
| 3f07994ed38e4c6cb32f349827d90f7c | admin |
97999e2533214d559cf09c94f86bfd75 | 6824420035fd463da599bc91fd5d213a |
+----------------------------------+-------+----------------------------+
```

21. Create the Keystone service and endpoint.

```
[root@rhos2 ~]# keystone service-create --name=keystone --type=identity
--description="Keystone Identity Service"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |    Keystone Identity Service     |
|      id     | 750b8753f1ef471c8952ece719fe806e |
|     name    |             keystone             |
|     type    |             identity             |
+-------------+----------------------------------+

[root@rhos2 ~]# keystone service-list
+----------------------------------+----------+----------+--------------+
|                id                |   name   |   type   |
description        |
+----------------------------------+----------+----------+--------------+
| 750b8753f1ef471c8952ece719fe806e | keystone | identity | Keystone
Identity Service |
+----------------------------------+----------+----------+--------------+

[root@rhos2 ~]# keystone endpoint-create --service keystone --publicurl
'http://rhos-keystone-vip:5000/v2.0' --adminurl 'http://rhos-keystone-
vip:35357/v2.0' --internalurl 'http://rhos-keystone-vip:5000/v2.0'
+-------------+-----------------------------------+
|   Property  |               Value               |
+-------------+-----------------------------------+
|   adminurl  | http://rhos-keystone-vip:35357/v2.0 |
|      id     |   1ce7d7175521409892e7f46fa5859691  |
| internalurl |  http://rhos-keystone-vip:5000/v2.0 |
|  publicurl  |  http://rhos-keystone-vip:5000/v2.0 |
|    region   |              regionOne              |
|  service_id |   750b8753f1ef471c8952ece719fe806e  |
+-------------+-----------------------------------+

[root@rhos2 ~]# keystone endpoint-list
+----------------------------------+----------+-----------------------------------+
|                id                |   region  |             publicurl
|           internalurl            |              adminurl
|           service_id             |
```

```
+----------------------------------+----------------------------------+
| 1ce7d7175521409892e7f46fa5859691 | regionOne | http://rhos-keystone-
vip:5000/v2.0 | http://rhos-keystone-vip:5000/v2.0 | http://rhos-
keystone-vip:35357/v2.0 | 750b8753f1ef471c8952ece719fe806e |
+----------------------------------+----------------------------------+
```

22. To test, unset the token and endpoint environment variables on a single node, then source */etc/keystonerc_admin*.

```
[root@rhos2 ~]# unset OS_SERVICE_ENDPOINT

[root@rhos2 ~]# unset OS_SERVICE_TOKEN

[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# env | grep OS_
OS_PASSWORD=keystonetest
OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

23. Verify that the environment variables are set properly and that the administrator can issue a successful Keystone command. Repeat this test on all nodes.

```
[root@rhos2 ~(keystone_admin)]# keystone user-list
+----------------------------------+-------+---------+-------+
|                id                |  name | enabled | email |
+----------------------------------+-------+---------+-------+
| 97999e2533214d559cf09c94f86bfd75 | admin |   True  |       |
+----------------------------------+-------+---------+-------+
```

24. View cluster status.

```
[root@rhos2 ~(keystone_admin)]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 11:42:57 2014
Last change: Thu Jun 12 11:27:35 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
12 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
```

```
      mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
      mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
      mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
      Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

25. Verify keystone is running on all cluster nodes.

```
[root@rhos2 ~(keystone_admin)]# ps -ef | grep keystone
keystone  8171     1  0 11:27 ?        00:00:02 /usr/bin/python
/usr/bin/keystone-all
root     10042  7710  0 11:43 pts/2    00:00:00 grep keystone
```

4.3.4 Install Memcached

Memcached is a distributed memory object-caching system designed to alleviate database load by storing arbitrary data in memory for quick retrieval. In this reference architecture, Memcached backs several OpenStack services including Cinder and Nova. This section of the paper describes how to install an active/active Memcached cluster on the controller nodes.

1. On all controller nodes, add the port for **memcached**, save, and restart the firewall.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 11211 -j ACCEPT

[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]
```

2. Install **memcached** on all controller nodes.

```
[root@rhos2 ~]# yum install -y memcached
```

3. On a single cluster node, create the active/active **memcached** cluster.

```
[root@rhos2 ~]# pcs resource create memcached lsb:memcached --clone
```

4. Verify that the cluster resource started successfully on all nodes.

```
[root@rhos2 ~]# pcs status | grep -A 1 memcached | grep Started | grep
rhos2 | grep rhos3 | grep rhos4
Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

5. Verify that **memcached** is running and that the cluster resource started successfully.

```
[root@rhos2 ~]# pcs status | grep -A 1 memcached | grep Started | grep
'rhos[2-4]'
Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

```
[root@rhos2 ~]# ps -ef | grep memcache
494      10493     1  0 11:45 ?        00:00:00 memcached -d -p 11211 -u
memcached -m 64 -c 1024 -P /var/run/memcached/memcached.pid
root     10506 10236  0 11:45 pts/2    00:00:00 grep memcache
```

6. Check status of the cluster.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 11:48:24 2014
Last change: Thu Jun 12 11:45:49 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
15 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

4.3.5 Implement HA Glance

OpenStack Glance provides services for discovering, registering, and retrieving virtual machine images. Glance can store images in a variety of locations. In this reference architecture Glance is configured as an active/active network. An NFS share mounted to all controller nodes acts as the image repository. This section of the reference architecture describes that steps used to configure Glance in the Red Hat Systems Engineering lab.

1. Add the Glance ports to all three controller nodes.

NOTE: Execute commands 1-5 on ALL controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 9191 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 9292 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep -E '9191|9292'
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:9292
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:9191
```

3. Install Glance on all three controller nodes.

```
[root@rhos2 ~]# yum install -y openstack-glance
```

4. On all controller nodes, configure the Glance API. These commands configure Glance to use Keystone, Qpid, and MySQL.

```
[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
DEFAULT sql_connection mysql://glance:glancetest@rhos-mysql-vip/glance

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
paste_deploy flavor keystone

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken auth_port 35357

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken auth_protocol http

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken admin_user glance

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
keystone_authtoken admin_password glancetest

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-api.conf
DEFAULT qpid_hostname rhos-qpid-vip
```

5. On all controller nodes, configure the Glance registry.

```
[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
```

```
DEFAULT sql_connection mysql://glance:glancetest@rhos-mysql-vip/glance

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
paste_deploy flavor keystone

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken auth_port 35357

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken auth_protocol http

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken admin_user glance

[root@rhos2 ~]# openstack-config --set /etc/glance/glance-registry.conf
keystone_authtoken admin_password glancetest
```

6. On a single controller node, source */root/keystonerc_admin* and create the Glance user.

   NOTE: Execute commands 6-15 on a single controller node.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# keystone user-create --name glance --pass
glancetest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | f820dffd3afb4ca38f313e1bca9a3456 |
|   name   |              glance              |
+----------+----------------------------------+
```

7. Add the Glance user to the admin role in the services tenant.

```
[root@rhos2 ~(keystone_admin)]# keystone user-role-add --user glance
--role admin --tenant services
```

8. Create the Glance service and endpoint.

```
[root@rhos2 ~(keystone_admin)]# keystone service-create --name glance
--type image --description "Glance Image Service"
+-------------+----------------------------------+
|   Property  |              Value               |
```

```
+-------------+----------------------------------+
| description |       Glance Image Service       |
|      id     | 11140bab74554d0e991c7b9aa6074597 |
|     name    |              glance              |
|     type    |              image               |
+-------------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone endpoint-create --service glance
--publicurl "http://rhos-glance-vip:9292" --adminurl "http://rhos-glance-
vip:9292" --internalurl "http://rhos-glance-vip:9292"

+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |   http://rhos-glance-vip:9292    |
|      id     | b52a89b8f3384d82a37bf0ac4fd57d39 |
| internalurl |   http://rhos-glance-vip:9292    |
|  publicurl  |   http://rhos-glance-vip:9292    |
|    region   |            regionOne             |
|  service_id | 11140bab74554d0e991c7b9aa6074597 |
+-------------+----------------------------------+
```

9. Create the Glance database.

```
[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "CREATE DATABASE glance;"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "GRANT ALL ON glance.* TO
'glance'@'%' IDENTIFIED BY 'glancetest';"

[root@rhos2 ~(keystone_admin)]# su glance -s /bin/sh -c "glance-manage
db_sync"
```

10. On a single controller, create a directory on the shared storage for the Glance repository.

```
[root@rhos2 ~(keystone_admin)]# mkdir -p /srv/rhos/glance
```

11. On a single controller, create the Glance file system cluster resource. It should NFS mount the shared file system to */var/lib/glance*.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create glance-fs Filesystem
device="nfs-server:/srv/rhos/glance"  directory="/var/lib/glance"
fstype="nfs" options="v4" --clone
```

12. Change ownership of */var/lib/glance* to the glance user.

```
[root@rhos2 ~(keystone_admin)]# chown glance:nobody /var/lib/glance

[root@rhos2 ~(keystone_admin)]# ls -al /var/lib | grep glance
drwxr-xr-x.  3 glance    nobody   4096 Apr 16 15:04 glance
```

NOTE: A Glance user with the same UID should exist on the NFS server and all cluster nodes. **Appendix B: Configure Storage Server** describes this step.

13. On a single controller create the Glance registry cluster resource.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create glance-registry
lsb:openstack-glance-registry --clone
```

14. On a single controller, create the Glance API cluster resource.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create glance-api
lsb:openstack-glance-api --clone
```

15. Set constraints on the Glance cluster resource start ordering so that the file system starts before the registry and the registry starts before the API service. All three services must be co-located.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint order start glance-fs-
clone then glance-registry-clone
Adding glance-fs-clone glance-registry-clone (kind: Mandatory) (Options:
first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add glance-
registry with glance-fs

[root@rhos2 ~(keystone_admin)]# pcs constraint order start glance-
registry-clone then glance-api-clone
Adding glance-registry-clone glance-api-clone (kind: Mandatory) (Options:
first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add glance-api
with glance-registry
```

16. Use **`pcs status`** to verify that all Glance services started correctly.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 11:57:44 2014
Last change: Thu Jun 12 11:50:25 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
24 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
```

```
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

17. View the resource constraints.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint show
Location Constraints:
Ordering Constraints:
  start glance-fs-clone then start glance-registry-clone
  start glance-registry-clone then start glance-api-clone
Colocation Constraints:
  glance-registry with glance-fs
  glance-api with glance-registry
```

18. Verify Glance is running on all controller nodes.

```
[root@rhos2 ~]# ps -ef | grep glance
glance   11470     1  0 11:50 ?        00:00:00 /usr/bin/python
/usr/bin/glance-registry
glance   11514 11470  0 11:50 ?        00:00:00 /usr/bin/python
/usr/bin/glance-registry
glance   11583     1  0 11:50 ?        00:00:01 /usr/bin/python
/usr/bin/glance-api
glance   11705 11583  0 11:50 ?        00:00:00 /usr/bin/python
/usr/bin/glance-api
root     13003 10898  0 11:57 pts/2    00:00:00 grep glance
```

19. Install the **`rhel-guest-image-6`** package.

```
[root@rhos2 ~]# yum install -y rhel-guest-image-6
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
```

20. Create a Glance image named *RHEL 6.5* from the image file.

```
[root@rhos2 ~]# glance image-create --name 'RHEL 6.5' --is-public true \
--disk-format qcow2 --container-format bare \
--file /usr/share/rhel-guest-image-6/rhel-guest-image-6-6.5-20140603.0-
1.qcow2
```

21. Verify the image status is active.

```
[root@rhos2 ~]# glance image-list
+--------------------------------------+----------+--------------------+
| ID                                   | Name     | Disk Format |
Container Format | Size      | Status |
+--------------------------------------+----------+--------------------+
| 6e087c2c-fb8f-4017-b677-18ae3b92118f | RHEL 6.5 | qcow2       | bare
| 342818816 | active |
+--------------------------------------+----------+--------------------+
```

4.3.6 Implement HA Cinder

Cinder is a block storage service for OpenStack. Cinder virtualizes pools of block storage and presents them to virtual machines. Self service users request block storage capacity via the Cinder API. They do not need to know the details of the underlying storage implementation. In this reference architecture Cinder is configured as an active/active HA cluster. An NFS share mounted to all controller nodes acts as the block storage repository. This section describes that steps to configure Cinder in the Red Hat Systems Engineering lab.

1. Add the Cinder ports to all three controller nodes.

NOTE: Perform steps 1-7 on ALL controller nodes.

```
[root@rhos2 ~(keystone_admin)]# iptables -I INPUT -p tcp -m tcp --dport
8776 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~(keystone_admin)]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~(keystone_admin)]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~(keystone_admin)]# iptables -L -n | grep 8776
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:8776
```

3. Install Cinder on all three controller nodes.

```
[root@rhos2 ~(keystone_admin)]# yum install -y openstack-cinder
```

4. On all controller nodes, configure Cinder to use Keystone, Glance, Qpid, Memcached, and MySQL.

```
[root@rhos2 ~(keystone_admin)]# openstack-config --set
```

```
/etc/cinder/cinder.conf DEFAULT auth_strategy keystone

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf keystone_authtoken auth_host rhos-keystone-vip

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf keystone_authtoken admin_tenant_name services

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf keystone_authtoken admin_user cinder

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf keystone_authtoken admin_password cindertest

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT rpc_backend
cinder.openstack.common.rpc.impl_qpid

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT qpid_hostname rhos-qpid-vip

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT qpid_port 5672

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT sql_connection
mysql://cinder:cindertest@rhos-mysql-vip/cinder

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT max_retries -1

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT retry_interval 1

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT glance_host rhos-glance-vip

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf keystone_authtoken memcached_servers rhos2-
priv:11211,rhos3-priv:11211,rhos4-priv:11211
```

5. Create the Cinder share on all controller nodes.

```
[root@rhos2 ~(keystone_admin)]# mkdir -p /srv/rhos/cinder
```

6. Specify the underlying NFS share for Cinder in */etc/cinder/nfs_exports*. In this example it is *nfs-server:/srv/rhos/cinder*. Verify the file is owned by the Cinder group and that the appropriate permissions are set. Perform this step on all controller nodes.

```
[root@rhos2 ~(keystone_admin)]# cat /etc/cinder/nfs_exports
nfs-server:/srv/rhos/cinder

[root@rhos2 ~(keystone_admin)]# chown root:cinder /etc/cinder/nfs_exports
```

```
[root@rhos2 ~(keystone_admin)]# chmod 0640 /etc/cinder/nfs_exports
```

7. Configure Cinder to use the NFS backend on all controller nodes.

```
[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT nfs_shares_config /etc/cinder/nfs_exports

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT nfs_sparsed_volumes true

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT nfs_mount_options v4

[root@rhos2 ~(keystone_admin)]# openstack-config --set
/etc/cinder/cinder.conf DEFAULT volume_driver
cinder.volume.drivers.nfs.NfsDriver
```

8. Create the Cinder user and associate it with the admin role.

NOTE: Perform steps 8-14 on a single controller node.

```
[root@rhos2 ~(keystone_admin)]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# keystone user-create --name cinder --pass
cindertest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 0fe0286c7e524fffbe1f7f016473dffd |
|   name   |              cinder              |
+----------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone user-role-add --user cinder
--role admin --tenant services

[root@rhos2 ~(keystone_admin)]# keystone user-role-list --user cinder
--tenant services
+----------------------------------+-------+----------------------------------+
|                id                |  name |             user_id
|            tenant_id             |
+----------------------------------+-------+----------------------------------+
| 3f07994ed38e4c6cb32f349827d90f7c | admin |
0fe0286c7e524fffbe1f7f016473dffd | 7ebb91e93b2c49118537f9ef999fe941 |
+----------------------------------+-------+----------------------------------+
```

9. Create the Cinder service and endpoint.

```
[root@rhos2 ~(keystone_admin)]# keystone service-create --name cinder
--type volume --description "Cinder Volume Service"
```

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |      Cinder Volume Service       |
|      id     | c90cf58dd4be4a4099b68db6443f83b4 |
|     name    |              cinder              |
|     type    |              volume              |
+-------------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone endpoint-create --service cinder
--publicurl "http://rhos-cinder-vip:8776/v1/\$(tenant_id)s" --adminurl
"http://rhos-cinder-vip:8776/v1/\$(tenant_id)s" --internalurl
"http://rhos-cinder-vip:8776/v1/\$(tenant_id)s"
+-------------+-------------------------------------------------+
|   Property  |                      Value                      |
+-------------+-------------------------------------------------+
|   adminurl  | http://rhos-cinder-vip:8776/v1/$(tenant_id)s |
|      id     |       7af6f733edb64d19b5065d013be21e2f       |
| internalurl | http://rhos-cinder-vip:8776/v1/$(tenant_id)s |
|  publicurl  | http://rhos-cinder-vip:8776/v1/$(tenant_id)s |
|    region   |                  regionOne                  |
|  service_id |       c90cf58dd4be4a4099b68db6443f83b4       |
+-------------+-------------------------------------------------+
```

10. Create the Cinder database on one controller node.

```
[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "CREATE DATABASE cinder;"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "GRANT ALL ON cinder.* TO 'cinder'@'%'
IDENTIFIED BY 'cindertest';"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "FLUSH PRIVILEGES;"

[root@rhos2 ~(keystone_admin)]# su cinder -s /bin/sh -c "cinder-manage db
sync"
```

11. Create the Cinder-API cluster resource group on one controller node.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create cinder-api
lsb:openstack-cinder-api --clone
```

12. Create the Cinder scheduler cluster resource group on one controller node.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create cinder-scheduler
lsb:openstack-cinder-scheduler --clone
```

13. Create the Cinder volume cluster resource group on one controller node.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create cinder-volume
lsb:openstack-cinder-volume --clone
```

14. Add the resource group co-location and start constraints. API should start before scheduler and scheduler before volume. Co-locate all three resources. Perform this step on one controller node.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint order start cinder-api-
clone then cinder-scheduler-clone
Adding cinder-api-clone cinder-scheduler-clone (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add cinder-
scheduler with cinder-api

[root@rhos2 ~(keystone_admin)]# pcs constraint order start cinder-
scheduler-clone then cinder-volume-clone
Adding cinder-scheduler-clone cinder-volume-clone (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add cinder-
volume with cinder-scheduler
```

15. Use **`pcs status`** to verify cluster resources are started on all controller nodes.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 12:04:44 2014
Last change: Thu Jun 12 12:02:49 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
33 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

```
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

16. View the running Cinder processes.

```
[root@rhos2 ~]# ps -ef | grep cinder
cinder   14066     1  2 12:02 ?        00:00:02 /usr/bin/python
/usr/bin/cinder-api --log-file /var/log/cinder/api.log
cinder   14200     1  1 12:02 ?        00:00:01 /usr/bin/python
/usr/bin/cinder-scheduler --config-file /usr/share/cinder/cinder-
dist.conf --config-file /etc/cinder/cinder.conf --logfile
/var/log/cinder/scheduler.log
cinder   14210     1  1 12:02 ?        00:00:02 /usr/bin/python
/usr/bin/cinder-volume --config-file /usr/share/cinder/cinder-dist.conf
--config-file /etc/cinder/cinder.conf --logfile
/var/log/cinder/volume.log
cinder   14233 14210  1 12:02 ?        00:00:01 /usr/bin/python
/usr/bin/cinder-volume --config-file /usr/share/cinder/cinder-dist.conf
--config-file /etc/cinder/cinder.conf --logfile
/var/log/cinder/volume.log
root     14759 13213  0 12:04 pts/2    00:00:00 grep cinder
```

17. To test Cinder, source */root/keystonerc_admin*.

```
[root@rhos2 ~(keystone_admin)]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# env | grep OS_
OS_PASSWORD=keystonetest
OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

18. Create a Cinder volume.

```
[root@rhos2 ~(keystone_admin)]# cinder create --display-name rhos2-vol 1
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2014-06-12T17:02:53.865177      |
| display_description |                 None                 |
|     display_name    |               rhos2-vol              |
|          id         | d0514760-fa0d-439b-956f-911987a786c1 |
|       metadata      |                  {}                  |
|         size        |                  1                   |
|     snapshot_id     |                 None                 |
```

```
|     source_volid     |                  None                  |
|        status        |                creating                |
|     volume_type      |                  None                  |
+----------------------+----------------------------------------+
```

19. Verify the volume is available.

```
[root@rhos2 ~]# cinder list | grep available | grep rhos2
| d0514760-fa0d-439b-956f-911987a786c1 | available |  rhos2-vol   |  1
|     None    |  false   |             |
```

20. Delete the volume before continuing.

```
[root@rhos2 ~(keystone_admin)]# cinder delete rhos2-vol

[root@rhos2 ~(keystone_admin)]# cinder list
+----+--------+--------------+------+-------------+----------+----------+
| ID | Status | Display Name | Size | Volume Type | Bootable | Attached
to |
+----+--------+--------------+------+-------------+----------+----------+
```

4.3.7 Implement HA Neutron

Neutron provides self-service networking for OpenStack. Optional plugins extend network capabilities allowing users to deploy rich networks and networking services. In this reference architecture Neutron runs as an active/passive cluster resource on the controller nodes. This section describes the steps to install the Neutron server.

Neutron can be configured many possible ways. In this reference architecture Neutron is configured to use the Open vSwitch plugin. This plugin provides software-defined bridges and ports and a method for mapping them to physical interface ports.

VLANs are used within Open vSwitch to define separate networks. Open vSwitch enforces tenant networks for OpenStack through VLANs. Packets tagged with a specific VLAN header are only available to other devices in the same VLAN even if they share the same physical network.

This reference architecture uses a provider network to map the external lab network to a Neutron network object. This allows instances within a tenant to communicate with devices on the lab network.

NOTE: Neutron software packages were installed in **4.3.1 Configure the Network Bridges**. The network bridges required by Neutron were also created in this section.

1. Add the Neutron port to all three controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 9696 -j ACCEPT
```

NOTE: Execute steps 1-5 on ALL controller nodes.

2. Save and restart **iptables** to verify the port was added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep 9696
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:9696
```

3. Configure Neutron to use Keystone, MySQL, and Qpid on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/api-paste.ini
filter:authtoken auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/api-paste.ini
filter:authtoken admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/neutron/api-paste.ini
filter:authtoken admin_user neutron

[root@rhos2 ~]# openstack-config --set /etc/neutron/api-paste.ini
filter:authtoken admin_password neutrontest

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
auth_strategy keystone

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_user neutron

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_password neutrontest

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
rpc_backend neutron.openstack.common.rpc.impl_qpid

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
qpid_hostname rhos-qpid-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf database
connection mysql://neutron:neutrontest@rhos-mysql-vip:3306/ovs_neutron
```

4. Configure Neutron to use the Open vSwitch plugin on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
core_plugin
neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2

[root@rhos2 ~]# ln -s
/etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini
/etc/neutron/plugin.ini

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini
securitygroup firewall_driver
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini database
sql_connection mysql://neutron:neutrontest@rhos-mysql-vip/ovs_neutron
```

5. Configure Neutron to use VLANs for tenant networks.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
enable_tunneling False

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
tenant_network_type vlan
```

6. Source */root/keystonerc_admin* to execute commands as the OpenStack administrator.

NOTE: Execute steps 6-12 on a single controller node.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# env | grep OS_
OS_PASSWORD=keystonetest
OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

7. One one controller node, create the Neutron service user and assign it to the admin role.

```
[root@rhos2 ~(keystone_admin)]# keystone user-create --name neutron
--pass neutrontest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 645c04a0242343bf83154e50ead6301f |
|   name   |             neutron              |
+----------+----------------------------------+
```

```
[root@rhos2 ~(keystone_admin)]# keystone user-role-add --user
neutron --role admin --tenant services

[root@rhos2 ~(keystone_admin)]# keystone user-role-list --user neutron
--tenant services
+----------------------------------+----------------------------------+
|                id                |  name |             user_id
|            tenant_id             |
+----------------------------------+----------------------------------+
| 3f07994ed38e4c6cb32f349827d90f7c | admin |
645c04a0242343bf83154e50ead6301f | 7ebb91e93b2c49118537f9ef999fe941 |
+----------------------------------+----------------------------------+
```

8. Create the Neutron service and endpoint on a single controller node.

```
[root@rhos2 ~(keystone_admin)]# keystone service-create --name neutron
--type network --description "OpenStack Networking Service"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |   OpenStack Networking Service   |
|      id     | 7fcd9ea195944621926d426a54958fea |
|     name    |             neutron              |
|     type    |             network              |
+-------------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone endpoint-create --service
neutron --publicurl "http://rhos-neutron-vip:9696" --adminurl
"http://rhos-neutron-vip:9696" --internalurl "http://rhos-neutron-
vip:9696"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |   http://rhos-neutron-vip:9696   |
|      id     | b7a29687c86d4f889bae70286a042286 |
| internalurl |   http://rhos-neutron-vip:9696   |
|  publicurl  |   http://rhos-neutron-vip:9696   |
|    region   |            regionOne             |
|  service_id | 7fcd9ea195944621926d426a54958fea |
+-------------+----------------------------------+
```

9. Create the Neutron database on a single controller node.

```
[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "CREATE DATABASE ovs_neutron;"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "GRANT ALL ON ovs_neutron.* TO 'neutron'@'%'
IDENTIFIED BY 'neutrontest';"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "FLUSH PRIVILEGES;"

[root@rhos2 ~(keystone_admin)]# neutron-db-manage  --config-file
```

```
/usr/share/neutron/neutron-dist.conf --config-file
/etc/neutron/neutron.conf --config-file /etc/neutron/plugin.ini stamp
havana
No handlers could be found for logger "neutron.common.legacy"
INFO  [alembic.migration] Context impl MySQLImpl.
INFO  [alembic.migration] Will assume non-transactional DDL.
```

10. On a single controller node, create the Neutron service and database check cluster resources.

```
[root@rhos2 ~(keystone_admin)]# pcs property set start-failure-is-
fatal=false

[root@rhos2 ~(keystone_admin)]# pcs resource create neutron-db-check
lsb:neutron-db-check meta failure-timeout=5 --clone

[root@rhos2 ~(keystone_admin)]# pcs resource create neutron-server
lsb:neutron-server
```

11. Create the Neutron server colocation and startup rules. The database check should precede the server. Both should be co-located.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint order start neutron-db-
check then neutron-server
Adding neutron-db-check neutron-server (kind: Mandatory) (Options: first-
action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add neutron-
server with neutron-db-check
```

12. Verify the constraint ordering.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint show
Location Constraints:
Ordering Constraints:
  start glance-fs-clone then start glance-registry-clone
  start glance-registry-clone then start glance-api-clone
  start cinder-api-clone then start cinder-scheduler-clone
  start cinder-scheduler-clone then start cinder-volume-clone
  start neutron-db-check then start neutron-server
Colocation Constraints:
  glance-registry with glance-fs
  glance-api with glance-registry
  cinder-scheduler with cinder-api
  cinder-volume with cinder-scheduler
  neutron-server with neutron-db-check
```

13. Clean up the Neutron services.

```
[root@rhos2 ~]# pcs resource cleanup neutron-server
Resource: neutron-server successfully cleaned up
```

14. View cluster status.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 12:11:17 2014
Last change: Thu Jun 12 12:07:32 2014 via crmd on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
37 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
```

15. Use **SSH** to view the running processes on the service-owning node. In this example **neutron-server** is started on rhos3.

```
[root@rhos2 ~]# ssh rhos3 ps -ef | grep neutron
neutron   9489     1  0 12:07 ?        00:00:01 /usr/bin/python
/usr/bin/neutron-server --config-file /usr/share/neutron/neutron-
dist.conf --config-file /etc/neutron/neutron.conf --config-file
/etc/neutron/plugin.ini --log-file /var/log/neutron/server.log
```

4.3.8 Implement HA Neutron Network Agents

Various agents support Neutron's core networking services. Among them are the DHCP agent, L3 agent, and Metadata agent.

- The DHCP agent assigns IP addresses to virtual machines via the **dnsmasq** utility. **dnsmasq** is a lightweight utility that provides network services such as DHCP and DNS for each tenant network.
- The L3 agent uses the Linux IP stack and **iptables** to perform NAT and L3 forwarding. This allows users and administrators to define software networks and routers. Linux network namespaces provide forwarding contexts for tenants with overlapping IP address ranges.
- The Metadata agent forwards virtual machine requests from tenant networks to the Nova metadata API. The Nova metadata service provides pre and post boot customization services to virtual machines.

In this reference architecture the Neutron network agents run as an active/passive HA cluster resource on the controller nodes. This section of the paper describes the steps used to configure the Neutron network agents in the Systems Engineering lab while validating this architecture.

1. Add the Neutron metadata agent and metadata proxy ports to all controller nodes.

   NOTE: Perform steps 1-7 on ALL controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 8775 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 9697 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep -E '8775|9697'
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:9697
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:8775
```

3. On all controller nodes, configure the network bridges for the Open vSwitch plugin. Verify that *local_ip* is set to the IP address of the external bridge interface.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
```

```
bridge_mappings physnet1:br-p3p1

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
enable_tunneling False

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
tenant_network_type vlan

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
network_vlan_ranges physnet1:1000:1010

[root@rhos2 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
integration_bridge br-int
```

4. Configure Neutron to support the metadata agent on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT auth_strategy keystone

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT auth_url http://rhos-keystone-vip:35357/v2.0

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT auth_region regionOne

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT admin_user neutron

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT admin_password neutrontest

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT nova_metadata_ip rhos-nova-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT nova_metadata_port 8775

[root@rhos2 ~]# openstack-config --set /etc/neutron/metadata_agent.ini
DEFAULT metadata_proxy_shared_secret metatest
```

5. Configure the Neutron DHCP agent to interface with Open vSwitch on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/dhcp_agent.ini
DEFAULT interface_driver neutron.agent.linux.interface.OVSInterfaceDriver
```

6. Configure the Neutron L3 agent on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
interface_driver neutron.agent.linux.interface.OVSInterfaceDriver

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
metadata_ip rhos-nova-vip

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
external_network_bridge ""

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
host rhos4-neutron-n

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
metadata_port 9697

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
handle_internal_only_routers True

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
use_namespaces True

[root@rhos2 ~]# openstack-config --set /etc/neutron/l3_agent.ini DEFAULT
enable_metadata_proxy True
```

NOTE: The DEFAULT host entry for the L3 service must be set to the same value on all controller nodes. This ensures that all controllers use the same network namespace in the event of a failover. In this example the common host value for all L3 agents is *rhos4-neutron-n.*

7. Configure Neutron to watch the agent services on all controller nodes. Ensure the L3 agent watch value includes the DEFAULT L3 agent host value specified in the previous step.

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT auth_strategy keystone

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT auth_url http://rhos-keystone-vip:35357/v2.0

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT auth_region regionOne

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT admin_user neutron

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT admin_password neutrontest
```

```
[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT watched_agents l3:/var/run/neutron/neutron-l3-
agent.pid:/var/run/neutron/neutron-l3-
agent.status:host=rhos4-neutron-n

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT check_interval 20

[root@rhos2 ~]# openstack-config --set /etc/neutron/agent_watch.ini
DEFAULT max_agent_downtime 3600
```

8. Create a Neutron database resource check in the *neutron-agents* resource group on a single controller node.

NOTE: Perform steps 8-16 on a single controller node.

```
[root@rhos2 ~]# pcs resource defaults resource-stickiness=100

[root@rhos2 ~]# pcs property set start-failure-is-fatal=false

[root@rhos2 ~]# pcs resource create neutron-db-check-pre lsb:neutron-db-
check meta failure-timeout=5 --group neutron-agents-pre
```

9. Add Pacemaker resources for Open vSwitch cleanup, Network namespaces cleanup, and a monitoring agent. These resources clean up after the Neutron agent in event of a failover. Perform these steps on a single controller node.

```
[root@rhos2 ~]# pcs resource create neutron-ovs-cleanup lsb:neutron-ovs-
cleanup --group neutron-agents-pre

[root@rhos2 ~]# pcs resource create neutron-netns-cleanup lsb:neutron-
netns-cleanup --group neutron-agents-pre

[root@rhos2 ~]# pcs resource create neutron-agent-watch lsb:neutron-
agent-watch --group neutron-agents-pre
```

10. On a single controller node, create the Neutron Open vSwitch agent.

```
[root@rhos2 ~]# pcs resource create neutron-openvswitch-agent
lsb:neutron-openvswitch-agent
```

11. On a single controller node, create active/passive cluster resources for the DHCP, L3, and metadata agents. Do not proceed until the cluster resources are online.

```
[root@rhos2 ~]# pcs resource create neutron-dhcp-agent lsb:neutron-dhcp-
agent

[root@rhos2 ~]# pcs resource create neutron-l3-agent lsb:neutron-l3-agent

[root@rhos2 ~]# pcs resource create neutron-metadata-agent lsb:neutron-
```

```
metadata-agent
```

12. Add the resource location and ordering constraints on a single controller node. The *neutron-agents-pre* resource group should proceed neutron Open vSwitch. Neutron Open vSwitch should proceed the DHCP, L3, and metadata agents. All of the resources and groups should be collocated.

```
[root@rhos2 ~]# pcs constraint order start neutron-agents-pre then
neutron-openvswitch-agent
Adding neutron-agents-pre neutron-openvswitch-agent (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~]# pcs constraint colocation add neutron-openvswitch-agent
with neutron-agents-pre

[root@rhos2 ~]# pcs constraint order start neutron-openvswitch-agent then
neutron-dhcp-agent
Adding neutron-openvswitch-agent neutron-dhcp-agent (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~]# pcs constraint colocation add neutron-dhcp-agent with
neutron-openvswitch-agent

[root@rhos2 ~]# pcs constraint order start neutron-openvswitch-agent then
neutron-l3-agent
Adding neutron-openvswitch-agent neutron-l3-agent (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~]# pcs constraint colocation add neutron-l3-agent with
neutron-openvswitch-agent

[root@rhos2 ~]# pcs constraint order start neutron-openvswitch-agent then
neutron-metadata-agent
Adding neutron-openvswitch-agent neutron-metadata-agent (kind: Mandatory)
(Options: first-action=start then-action=start)

[root@rhos2 ~]# pcs constraint colocation add neutron-metadata-agent with
neutron-openvswitch-agent
```

13. Verify the constraint ordering.

```
[root@rhos2 ~]# pcs constraint show
Location Constraints:
Ordering Constraints:
  start glance-fs-clone then start glance-registry-clone
  start glance-registry-clone then start glance-api-clone
  start cinder-api-clone then start cinder-scheduler-clone
  start cinder-scheduler-clone then start cinder-volume-clone
  start neutron-db-check then start neutron-server
  start neutron-agents-pre then start neutron-openvswitch-agent
  start neutron-openvswitch-agent then start neutron-dhcp-agent
  start neutron-openvswitch-agent then start neutron-l3-agent
  start neutron-openvswitch-agent then start neutron-metadata-agent
```

```
Colocation Constraints:
  glance-registry with glance-fs
  glance-api with glance-registry
  cinder-scheduler with cinder-api
  cinder-volume with cinder-scheduler
  neutron-server with neutron-db-check
  neutron-openvswitch-agent with neutron-agents-pre
  neutron-dhcp-agent with neutron-openvswitch-agent
  neutron-l3-agent with neutron-openvswitch-agent
  neutron-metadata-agent with neutron-openvswitch-agent
```

14. Verify the resources started. In this example the Neutron agents are started on *rhos4*.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 12:17:09 2014
Last change: Thu Jun 12 12:13:10 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
45 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
```

```
 Resource Group: neutron-agents-pre
  neutron-db-check-pre  (lsb:neutron-db-check):     Started rhos4-priv
  neutron-ovs-cleanup   (lsb:neutron-ovs-cleanup):  Started rhos4-priv
  neutron-netns-cleanup (lsb:neutron-netns-cleanup):Started rhos4-priv
  neutron-agent-watch   (lsb:neutron-agent-watch):  Started rhos4-priv
  neutron-openvswitch-agent (lsb:neutron-openvswitch-agent) Started rhos4-
priv
  neutron-dhcp-agent    (lsb:neutron-dhcp-agent):   Started rhos4-priv
  neutron-l3-agent      (lsb:neutron-l3-agent):     Started rhos4-priv
  neutron-metadata-agent (lsb:neutron-metadata-agent): Started rhos4-priv
```

15. List the Neutron agents. All agents should be alive with *admin_state_up* reporting *True*.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# neutron agent-list
+--------------------------------------+-------+-------------------------+
| id                                   | agent_type         | host
| alive | admin_state_up |
+--------------------------------------+-------+-------------------------+
| bca7041b-67de-4e72-8cdf-e1ef261ea191 | DHCP agent         |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| c1dd62ca-a745-4774-aae3-ede0acb7b7a1 | L3 agent           | rhos4-
neutron-n                    | :-)   | True           |
| fe3db77c-e63b-4852-83e7-315135412adf | Open vSwitch agent |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
+--------------------------------------+-------+-------------------------+
```

16. SSH to the agent-owning controller and list Neutron processes. In this example the agents started on *rhos4*.

```
[root@rhos2 ~(keystone_admin)]# ssh rhos4 ps -ef | grep neutron
neutron  10087     1  0 12:13 ?        00:00:01 /usr/bin/python
/usr/bin/neutron-agent-watch --log-file /var/log/neutron/agent-watch.log
--config-file /usr/share/neutron/neutron-dist.conf --config-file
/etc/neutron/neutron.conf --config-file /etc/neutron/agent_watch.ini
neutron  10148     1  0 12:13 ?        00:00:01 /usr/bin/python
/usr/bin/neutron-openvswitch-agent --log-file
/var/log/neutron/openvswitch-agent.log --config-file
/usr/share/neutron/neutron-dist.conf --config-file
/etc/neutron/neutron.conf --config-file
/etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini
neutron  10179     1  0 12:13 ?        00:00:00 /usr/bin/python
/usr/bin/neutron-metadata-agent --log-file /var/log/neutron/metadata-
agent.log --config-file /usr/share/neutron/neutron-dist.conf --config-
file /etc/neutron/neutron.conf --config-file
/etc/neutron/metadata_agent.ini
neutron  10180     1  0 12:13 ?        00:00:02 /usr/bin/python
/usr/bin/neutron-l3-agent --log-file /var/log/neutron/l3-agent.log
--config-file /usr/share/neutron/neutron-dist.conf --config-file
/etc/neutron/neutron.conf --config-file /etc/neutron/l3_agent.ini
--config-file /etc/neutron/fwaas_driver.ini
neutron  10181     1  0 12:13 ?        00:00:01 /usr/bin/python
/usr/bin/neutron-dhcp-agent --log-file /var/log/neutron/dhcp-agent.log
```

```
--config-file /usr/share/neutron/neutron-dist.conf --config-file
/etc/neutron/neutron.conf --config-file /etc/neutron/dhcp_agent.ini
root     10313 10148  0 12:13 ?        00:00:00 sudo neutron-rootwrap
/etc/neutron/rootwrap.conf ovsdb-client monitor Interface name
--format=json
root     10315 10313  0 12:13 ?        00:00:00 /usr/bin/python
/usr/bin/neutron-rootwrap /etc/neutron/rootwrap.conf ovsdb-client monitor
Interface name --format=json
```

4.3.9 Implement HA Nova

Nova is OpenStack's cloud computing fabric controller. The core Nova services described in this reference architecture include:

- API – the endpoint for Nova service requests
- Conductor – this service allows OpenStack compute nodes to function without directly accessing the state database in order to improve security.
- Scheduler – a service that schedules Nova service requests to the appropriate resources
- VNC console proxy – this service enables users to access their virtual machines through VNC clients
- Metadata service – enables virtual machines to retrieve instance-specific data

In this reference architecture the Nova API, Scheduler, VNC console, and Metadata services reside on the HA controller nodes. All of these services run as active/active HA cluster resources with the exception of Nova scheduler, which runs as an active/passive resource. This section of the reference architecture describes the steps followed to configure Nova on the controller nodes in the Systems Engineering Lab.

1. Add the Nova service ports to all controller nodes.

NOTE: Perform steps 1-8 on all controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 6080 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 6081 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 8774 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
```

```
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep -E '6080|6081|8774'
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:8774
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:6081
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:6080
```

3. Install Nova software on all controller nodes.

```
[root@rhos2 ~]# yum install -y openstack-nova-console openstack-nova-
novncproxy openstack-nova-api openstack-nova-conductor openstack-nova-
scheduler
```

4. Configure Nova to use MySQL, Qpid, and Memcached on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
memcached_servers rhos2-priv:11211,rhos3-priv:11211,rhos4-priv:1211

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
sql_connection mysql://nova:rhos-mysql-vip/nova

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
auth_strategy keystone

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
rpc_backend nova.openstack.common.rpc.impl_qpid

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
qpid_hostname rhos-qpid-vip
```

5. Configure the VNC console proxy on all controller nodes.

```
[root@rhos2 ~]# openstack-config  --set /etc/nova/nova.conf DEFAULT
vncserver_proxyclient_address $(ip addr show dev em1 scope global | grep
inet | sed -e 's#.*inet ##g' -e  's#/.*##g')

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
vncserver_listen 0.0.0.0

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
novncproxy_base_url
http://rhos2.cloud.lab.eng.bos.redhat.com:6080/vnc_auto.html
```

6. Configure the Nova metadata service on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_host rhos-nova-vip

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_listen 0.0.0.0

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_listen_port 8775
```

```
[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
service_neutron_metadata_proxy True
```

7. Configure Nova to use Neutron networking on all controller nodes.

```
[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_metadata_proxy_shared_secret metatest

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
glance_host rhos-glance-vip

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
network_api_class nova.network.neutronv2.api.API

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_url http://rhos-neutron-vip:9696/

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_username neutron

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_password neutrontest

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_auth_url http://rhos-keystone-vip:35357/v2.0

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
firewall_driver nova.virt.firewall.NoopFirewallDriver

[root@rhos2 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
libvirt_vif_driver nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
```

8. Add Nova authentication information to */etc/nova/api-paste.ini*. This middleware configuration file is used by the Openstack Compute API and EC2 API.

```
[root@rhos2 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken auth_host rhos-keystone-vip

[root@rhos2 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_tenant_name services

[root@rhos2 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_user compute

[root@rhos2 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_password novatest
```

9. On a single controller node create the Keystone entries for the Nova user as the OpenStack administrator.

NOTE: Perform steps 9-15 on a single controller node.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# keystone user-create --name compute
--pass novatest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | e47f2071075744678a7f31699e92a040 |
|   name   |             compute              |
+----------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone user-role-add --user compute
--role admin --tenant services

[root@rhos2 ~(keystone_admin)]# keystone user-role-list --user compute
--tenant services
+----------------------------------+----------------------------------+
|                id                |  name |             user_id
|            tenant_id             |
+----------------------------------+----------------------------------+
| 3f07994ed38e4c6cb32f349827d90f7c | admin |
e47f2071075744678a7f31699e92a040 | 7ebb91e93b2c49118537f9ef999fe941 |
+----------------------------------+----------------------------------+
```

10. On a single controller node create the Nova service and endpoint. The virtual IP for the service acts as the service endpoint.

```
[root@rhos2 ~(keystone_admin)]# keystone service-create --name compute
--type compute --description "OpenStack Compute Service"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |    OpenStack Compute Service     |
|      id     | d91828e29dcd4acfb1b4bed1d4329b77 |
|     name    |             compute              |
|     type    |             compute              |
+-------------+----------------------------------+

[root@rhos2 ~(keystone_admin)]# keystone endpoint-create  --service
compute --publicurl "http://rhos-nova-vip:8774/v2/\$(tenant_id)s"
--adminurl "http://rhos-nova-vip:8774/v2/\$(tenant_id)s" --internalurl
"http://rhos-nova-vip:8774/v2/\$(tenant_id)s"
+-------------+-----------------------------------------+
|   Property  |                  Value                  |
+-------------+-----------------------------------------+
|   adminurl  | http://rhos-nova-vip:8774/v2/$(tenant_id)s |
|      id     |      1ffac066620b4fe3be4be84f88c67669      |
```

```
| internalurl | http://rhos-nova-vip:8774/v2/$(tenant_id)s |
|  publicurl  | http://rhos-nova-vip:8774/v2/$(tenant_id)s |
|    region   |                 regionOne                  |
|  service_id |      d91828e29dcd4acfb1b4bed1d4329b77      |
+-------------+--------------------------------------------+
```

11. On a single controller node create the Nova database.

```
[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "CREATE DATABASE nova;"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "GRANT ALL ON nova.* TO 'nova'@'%' IDENTIFIED BY
'novatest';"

[root@rhos2 ~(keystone_admin)]# mysql --user=root --password=redhat
--host=rhos-mysql-vip -e "FLUSH PRIVILEGES;"

[root@rhos2 ~(keystone_admin)]# su nova -s /bin/sh -c "nova-manage db
sync"
```

12. Create a Pacemaker resource for each Nova service on a single controller node. The *nova-scheduler* resource is active/passive. The rest are active/active.

```
[root@rhos2 ~(keystone_admin)]# pcs resource create nova-consoleauth
lsb:openstack-nova-consoleauth --clone

[root@rhos2 ~(keystone_admin)]# pcs resource create nova-novncproxy
lsb:openstack-nova-novncproxy --clone

[root@rhos2 ~(keystone_admin)]# pcs resource create nova-api
lsb:openstack-nova-api --clone

[root@rhos2 ~(keystone_admin)]# pcs resource create nova-scheduler
lsb:openstack-nova-scheduler

[root@rhos2 ~(keystone_admin)]# pcs resource create nova-conductor
lsb:openstack-nova-conductor --clone
```

13. Do not proceed until the Nova services are online.

```
[root@rhos2 ~]# pcs status | grep -A 1 nova-conductor | grep Started
Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

14. Add the resource location and ordering constraints on a single controller node. **`Nova-consoleauth`** should proceed **`novncproxy`**. **`Novncproxy`** should proceed the API. The API should proceed the scheduler. The scheduler should proceed the conductor service. The API, scheduler, consoleauth, and novncproxy services should be co-located.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint order start nova-
consoleauth-clone then nova-novncproxy-clone
Adding nova-consoleauth-clone nova-novncproxy-clone (kind: Mandatory)
```

```
(Options: first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add nova-
novncproxy with nova-consoleauth

[root@rhos2 ~(keystone_admin)]# pcs constraint order start nova-
novncproxy-clone then nova-api-clone
Adding nova-novncproxy-clone nova-api-clone (kind: Mandatory) (Options:
first-action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add nova-api
with nova-novncproxy

[root@rhos2 ~(keystone_admin)]# pcs constraint order start nova-api-clone
then nova-scheduler
Adding nova-api-clone nova-scheduler (kind: Mandatory) (Options: first-
action=start then-action=start)

[root@rhos2 ~(keystone_admin)]# pcs constraint colocation add nova-
scheduler with nova-api

[root@rhos2 ~(keystone_admin)]# pcs constraint order start nova-scheduler
then nova-conductor-clone
Adding nova-scheduler nova-conductor-clone (kind: Mandatory) (Options:
first-action=start then-action=start)
```

15. Verify the startup and colocation constraints.

```
[root@rhos2 ~(keystone_admin)]# pcs constraint show
Location Constraints:
Ordering Constraints:
  start glance-fs-clone then start glance-registry-clone
  start glance-registry-clone then start glance-api-clone
  start cinder-api-clone then start cinder-scheduler-clone
  start cinder-scheduler-clone then start cinder-volume-clone
  start neutron-db-check then start neutron-server
  start neutron-agents-pre then start neutron-openvswitch-agent
  start neutron-openvswitch-agent then start neutron-dhcp-agent
  start neutron-openvswitch-agent then start neutron-l3-agent
  start neutron-openvswitch-agent then start neutron-metadata-agent
  start nova-consoleauth-clone then start nova-novncproxy-clone
  start nova-novncproxy-clone then start nova-api-clone
  start nova-api-clone then start nova-scheduler
  start nova-scheduler then start nova-conductor-clone
Colocation Constraints:
  glance-registry with glance-fs
  glance-api with glance-registry
  cinder-scheduler with cinder-api
  cinder-volume with cinder-scheduler
  neutron-server with neutron-db-check
  neutron-openvswitch-agent with neutron-agents-pre
  neutron-dhcp-agent with neutron-openvswitch-agent
  neutron-l3-agent with neutron-openvswitch-agent
  neutron-metadata-agent with neutron-openvswitch-agent
```

```
nova-novncproxy with nova-consoleauth
nova-api with nova-novncproxy
nova-scheduler with nova-api
```

16. Test Nova functionality with the **nova usage** command.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~]# nova usage
Usage from 2014-05-15 to 2014-06-13:
None

[root@rhos2 ~]# nova usage-list
Usage from 2014-05-15 to 2014-06-13:
+-----------+---------+--------------+-----------+---------------+
| Tenant ID | Servers | RAM MB-Hours | CPU Hours | Disk GB-Hours |
+-----------+---------+--------------+-----------+---------------+
+-----------+---------+--------------+-----------+---------------+
```

17. View the running Nova services.

```
[root@rhos2 ~]# ps -ef | grep nova
nova     18495     1  1 12:19 ?        00:00:02 /usr/bin/python
/usr/bin/nova-consoleauth --logfile /var/log/nova/consoleauth.log
nova     18518     1  0 12:19 ?        00:00:00 /usr/bin/python
/usr/bin/nova-novncproxy --web /usr/share/novnc/
nova     18548     1  2 12:19 ?        00:00:04 /usr/bin/python
/usr/bin/nova-api --log-file /var/log/nova/api.log
nova     18586 18548  0 12:19 ?        00:00:00 /usr/bin/python
/usr/bin/nova-api --log-file /var/log/nova/api.log
nova     18587 18548  0 12:19 ?        00:00:01 /usr/bin/python
/usr/bin/nova-api --log-file /var/log/nova/api.log
nova     18588 18548  0 12:19 ?        00:00:00 /usr/bin/python
/usr/bin/nova-api --log-file /var/log/nova/api.log
nova     18609     1  0 12:19 ?        00:00:01 /usr/bin/python
/usr/bin/nova-conductor --logfile /var/log/nova/conductor.log
root     19474 18011  0 12:23 pts/2    00:00:00 grep nova
```

18. View cluster status.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 12:23:28 2014
Last change: Thu Jun 12 12:20:07 2014 via cibadmin on rhos3-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
58 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:
```

```
 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM): Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
 Resource Group: neutron-agents-pre
     neutron-db-check-pre (lsb:neutron-db-check):  Started rhos4-priv
     neutron-ovs-cleanup (lsb:neutron-ovs-cleanup): Started rhos4-priv
     neutron-netns-cleanup (lsb:neutron-netns-cleanup): Started rhos4-
priv
     neutron-agent-watch (lsb:neutron-agent-watch): Started rhos4-priv
 neutron-openvswitch-agent (lsb:neutron-openvswitch-agent) Started rhos4-
priv
 neutron-dhcp-agent    (lsb:neutron-dhcp-agent):   Started rhos4-priv
 neutron-l3-agent     (lsb:neutron-l3-agent):     Started rhos4-priv
 neutron-metadata-agent (lsb:neutron-metadata-agent) Started rhos4-priv
 Clone Set: nova-consoleauth-clone [nova-consoleauth]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-novncproxy-clone [nova-novncproxy]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-api-clone [nova-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 nova-scheduler  (lsb:openstack-nova-scheduler):  Started rhos3-priv
 Clone Set: nova-conductor-clone [nova-conductor]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

19. List the Nova services. All services should be enabled and up.

```
[root@rhos2 ~]# source /root/keystonerc_admin

[root@rhos2 ~(keystone_admin)]# nova service-list
```

```
+------------------+-----------------------------------+--------------+
| Binary           | Host                              | Zone     |
Status  | State | Updated_at                | Disabled Reason |
+------------------+-----------------------------------+--------------+
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:43.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:43.000000 | None            |
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:43.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:36.000000 | None            |
| nova-scheduler   | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:36.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:36.000000 | None            |
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:23:36.000000 | None            |
+------------------+-----------------------------------+--------------+
```

4.3.10 Implement HA Horizon

Horizon provides a web-based user interface to OpenStack services. In this reference architecture Horizon runs as an active/active HA cluster resource on all controller nodes. This section describes the steps taken to cluster Horizon in the Systems Engineering lab.

NOTE: In this reference architecture, the HAproxy load balancer was not configured with SSL support. Therefore all connections to Horizon must be made via HTTP.

1. Add the Horizon service ports to all controller nodes.

```
[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 80 -j ACCEPT

[root@rhos2 ~]# iptables -I INPUT -p tcp -m tcp --dport 443 -j ACCEPT
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos2 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos2 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: mangle nat filte[  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]

[root@rhos2 ~]# iptables -L -n | grep -E '80|443'
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:443
ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           tcp dpt:80
```

3. Set the SELinux Boolean to allow HTTP connections

```
[root@rhos2 ~]# setsebool -P httpd_can_network_connect on

[root@rhos2 ~]# getsebool httpd_can_network_connect
httpd_can_network_connect --> on
```

4. Install Horizon software on all controller nodes.

```
[root@rhos2 ~]# yum install -y mod_wsgi httpd mod_ssl openstack-dashboard
elinks
```

5. Change the dashboard settings to use **memcached** and allow connections to the virtual IP on all controller nodes.

```
[root@rhos2 ~]# sed  -i -e "s#ALLOWED_HOSTS.*#ALLOWED_HOSTS = ['*',]#g"
-e  "s#^CACHES#SESSION_ENGINE =
'django.contrib.sessions.backends.cache'\nCACHES#g#" -e
"s#locmem.LocMemCache'#memcached.MemcachedCache',\n\t'LOCATION' :
[  'rhos2-priv:11211', 'rhos3-priv:11211', 'rhos4-priv:11211', ]#g" -e
's#OPENSTACK_HOST =.*#OPENSTACK_HOST = "rhos-keystone-vip"#g'
/etc/openstack-dashboard/local_settings
```

6. Configure HTTP access permissions on all controller nodes in */etc/httpd/conf/server-status.conf*.

```
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from localhost
</Location>
```

7. Start and stop **httpd** on one controller node to generate a new secret key store.

```
[root@rhos2 ~]# service httpd start
Starting httpd: httpd: Could not reliably determine the server's fully
qualified domain name, using rhos2.cloud.lab.eng.bos.redhat.com for
ServerName
                                                           [  OK  ]
[root@rhos2 ~]# service httpd stop
Stopping httpd:                                            [  OK  ]
```

8. Copy the secret key store to the other controller nodes.

```
[root@rhos2 ~]# rsync -zvr /var/lib/openstack-dashboard/.secret_key_store
rhos3:/var/lib/openstack-dashboard/.secret_key_store-new

sending incremental file list
.secret_key_store

sent 144 bytes  received 31 bytes  50.00 bytes/sec
total size is 64  speedup is 0.37
```

```
[root@rhos2 ~]# rsync -zvr /var/lib/openstack-dashboard/.secret_key_store
rhos4:/var/lib/openstack-dashboard/.secret_key_store-new

sending incremental file list
.secret_key_store

sent 144 bytes  received 31 bytes  50.00 bytes/sec
total size is 64  speedup is 0.37
```

9. Overwrite the original secret key store with the new secret key store on the other controller nodes.

```
[root@rhos3 ~]# mv --force /var/lib/openstack-
dashboard/.secret_key_store-new /var/lib/openstack-
dashboard/.secret_key_store

[root@rhos4 ~]# mv --force /var/lib/openstack-
dashboard/.secret_key_store-new /var/lib/openstack-
dashboard/.secret_key_store
```

10. Set the apache user and group as the secret key store owner on all controller nodes.

```
[root@rhos2 ~]# chown apache:apache /var/lib/openstack-
dashboard/.secret_key_store

[root@rhos2 ~]# ls -al /var/lib/openstack-dashboard/.secret_key_store
-rw-------. 1 apache apache 64 Apr 29 13:07 /var/lib/openstack-
dashboard/.secret_key_store
```

11. On a single controller node create the Apache cluster resource.

```
[root@rhos2 ~]# pcs resource create httpd apache --clone
```

12. Verify httpd service is running.

```
[root@rhos2 ~]# ps -ef | grep http
root     20117     1  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20135 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20136 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20137 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20138 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20139 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20140 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20141 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20142 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
apache   20143 20117  0 12:25 ?        00:00:00 /usr/sbin/httpd
root     20565 19762  0 12:27 pts/2    00:00:00 grep http
```

13. Verify the resource started correctly on all controller nodes.

```
[root@rhos2 ~]# pcs status
Cluster name: osp4-controller
```

```
Last updated: Thu Jun 12 12:28:58 2014
Last change: Thu Jun 12 12:26:56 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
61 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm     (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
 Resource Group: neutron-agents-pre
     neutron-db-check-pre   (lsb:neutron-db-check):     Started rhos4-
priv
     neutron-ovs-cleanup    (lsb:neutron-ovs-cleanup):  Started rhos4-
priv
     neutron-netns-cleanup  (lsb:neutron-netns-cleanup):Started rhos4-
priv
     neutron-agent-watch    (lsb:neutron-agent-watch):  Started rhos4-
priv
 neutron-openvswitch-agent  (lsb:neutron-openvswitch-agent):  Started
rhos4-priv
 neutron-dhcp-agent    (lsb:neutron-dhcp-agent):   Started rhos4-priv
 neutron-l3-agent      (lsb:neutron-l3-agent):     Started rhos4-priv
 neutron-metadata-agent     (lsb:neutron-metadata-agent) Started rhos4-
priv
```

```
 Clone Set: nova-consoleauth-clone [nova-consoleauth]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-novncproxy-clone [nova-novncproxy]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-api-clone [nova-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 nova-scheduler  (lsb:openstack-nova-scheduler):   Started rhos3-priv
 Clone Set: nova-conductor-clone [nova-conductor]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: httpd-clone [httpd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

14. Access the dashboard via eLinks to test.

```
[root@rhos2 ~]# elinks --dump http://rhos-horizon-vip/dashboard
  Log In

    User Name
    [1]_____________________
    Password
    [2]_____________________
    [3]Sign In

References

    Visible links
```

15. Check for SELinux errors.

```
[root@rhos2 ~]# ausearch -m avc -i; audit2allow -al
<no matches>
```

4.4 Configure the Compute Nodes

OpenStack compute nodes provide a KVM hypervisor to launch virtual machines, the Nova compute service to accept Nova scheduler requests, and an Open vSwitch agent to participate in software defined networks. Nova compute nodes scale natively. The Nova scheduler load balances virtual machine requests across all registered compute nodes.

In this reference architecture the compute nodes are deployed as single node clusters with a Corosync ring size of 1. Pacemaker is installed on all compute nodes to monitor their services and recover from service failures. Pacemaker allows the compute nodes to recover from faults without user intervention which helps to improve availability.

This section of the paper describes how the single node clusters were deployed in the Systems Engineering lab for this reference architecture.

> **NOTE:** Repeat these steps for each compute node. Three compute nodes were deployed in this reference architecture. These steps can be performed in parallel or serially on each compute node.

1. Configure the Nova compute node **iptables** rules.

```
[root@rhos7 ~]#  iptables -F

[root@rhos7 ~]#  iptables -A INPUT -p tcp -m multiport --dports 5900:5999
-j ACCEPT

[root@rhos7 ~]# iptables -A INPUT -m state --state RELATED,ESTABLISHED -j
ACCEPT

[root@rhos7 ~]# iptables -A INPUT -p icmp -j ACCEPT

[root@rhos7 ~]  iptables -A INPUT -i lo -j ACCEPT

[root@rhos7 ~]# iptables -A INPUT -p tcp -m state --state NEW -m tcp
--dport 22 -j ACCEPT

[root@rhos7 ~]# iptables -A INPUT -p tcp -m state --state NEW -m tcp
--dport 443 -j ACCEPT

[root@rhos7 ~]# iptables -A INPUT -j REJECT --reject-with icmp-host-
prohibited
```

2. Save and restart **iptables** to verify the ports were added successfully.

```
[root@rhos7 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]

[root@rhos7 ~]# service iptables restart
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]
[  OK  ]
```

3. Add the HA, LB, and OST RHN channels.

```
[root@rhos7 ~]# rhn-channel --user admin --password examplepasswd -a -c
rhel-x86_64-server-ha-6 -c rhel-x86_64-server-lb-6

[root@rhos7 ~]# rhn-channel -l
rhel-x86_64-server-6
rhel-x86_64-server-6-ost-4
rhel-x86_64-server-ha-6
rhel-x86_64-server-lb-6
```

4. Verify the **iproute** package is up to date.

NOTE: The **iproute** package must be at version 2.6.32-130 or higher. The package name should include **netns**.

```
[root@rhos7 ~]# yum update -y iproute
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.

[root@rhos7 ~]# rpm -q iproute
iproute-2.6.32-130.el6ost.netns.3.x86_64
```

5. Install Nova software.

```
[root@rhos7 ~]# yum install -y openstack-nova-compute openstack-utils
python-cinder openstack-neutron-openvswitch pacemaker fence-agents
resource-agents pcs cman ccs
This system is not registered to Red Hat Subscription Management. You can
use subscription-manager to register.
Stopping kdump:[  OK  ]
Starting kdump:[  OK  ]
```

6. Set the *virt_use_nfs* SELinux Boolean. This value is required to access instance volumes on NFS shares. For more information see the **Red Hat Enterprise Linux OpenStack Platform 4 User's Guide**.

```
[root@rhos7 ~]# setsebool -P virt_use_nfs on

[root@rhos7 ~]# getsebool virt_use_nfs
virt_use_nfs --> on
```

7. Start and enable Open vSwitch.

```
[root@rhos7 ~]# chkconfig openvswitch on

[root@rhos7 ~]# service openvswitch start
/etc/openvswitch/conf.db does not exist ... (warning).
Creating empty database /etc/openvswitch/conf.db           [  OK  ]
Starting ovsdb-server                                      [  OK  ]
Configuring Open vSwitch system IDs                        [  OK  ]
Inserting openvswitch module                               [  OK  ]
Starting ovs-vswitchd                                      [  OK  ]
Enabling remote OVSDB managers                             [  OK  ]
```

8. Configure the network initialization files to create and enable the bridge interfaces.

```
[root@rhos7 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-br-p3p1 << EOF
> DEVICE=br-p3p1
> DEVICETYPE=ovs
> TYPE=OVSBridge
> ONBOOT=yes
> OVSBOOTPROTO=static
> EOF

[root@rhos7 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-p3p1 << EOF
> DEVICE=p3p1
> NM_CONTROLLED=no
```

```
> ONBOOT=yes
> DEVICETYPE=ovs
> OVS_BRIDGE=br-p3p1
> TYPE=OVSPort
> EOF

[root@rhos7 ~]# cat > /etc/sysconfig/network-scripts/ifcfg-br-int << EOF
> DEVICE=br-int
> DEVICETYPE=ovs
> TYPE=OVSBridge
> ONBOOT=yes
> BOOTPROTO=none
> EOF
```

9. Restart networking to activate the bridges.

```
[root@rhos7 ~]# service network restart
Shutting down interface em1:                               [  OK  ]
Shutting down interface em2:                               [  OK  ]
Shutting down interface p3p1:                              [  OK  ]
Shutting down interface p3p2:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface br-int:                              [  OK  ]
Bringing up interface br-p3p1:                             [  OK  ]
Bringing up interface em1:
Determining IP information for em1... done.
                                                           [  OK  ]
Bringing up interface em2:  Determining if ip address 172.16.2.107 is
already in use for device em2...
                                                           [  OK  ]
Bringing up interface p3p1:  RTNETLINK answers: File exists
                                                           [  OK  ]
Bringing up interface p3p2:  Determining if ip address 172.31.139.107 is
already in use for device p3p2...
                                                           [  OK  ]
```

10. Verify that the bridges started.

```
[root@rhos7 ~]# ifconfig -a |  grep br
br-int    Link encap:Ethernet  HWaddr 9A:64:B4:C7:4B:49
br-p3p1   Link encap:Ethernet  HWaddr E0:DB:55:74:40:E5

[root@rhos2 ~]# ovs-vsctl show
3dbbf4bf-f637-46eb-b788-ae05bfc2488f
    Bridge br-int
        Port br-int
            Interface br-int
                type: internal
    Bridge "br-p3p1"
        Port "br-p3p1"
            Interface "br-p3p1"
                type: internal
        Port "p3p1"
```

```
            Interface "p3p1"
    ovs_version: "1.11.0"
```

11. Configure Nova to use Memcached, MySQL, Qpid, Glance, and Neutron

```
[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
memcached_servers rhos2-priv:11211,rhos3-priv:11211

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
sql_connection mysql://nova:novatest@rhos-mysql-vip/nova

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
auth_strategy keystone

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
rpc_backend nova.openstack.common.rpc.impl_qpid

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
qpid_hostname rhos-qpid-vip

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_host rhos-nova-vip

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_listen 0.0.0.0

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
metadata_listen_port 8775

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
service_neutron_metadata_proxy True

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_metadata_proxy_shared_secret metatest

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
glance_host rhos-glance-vip

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
network_api_class nova.network.neutronv2.api.API

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_url http://rhos-neutron-vip:9696/

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_tenant_name services

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_username neutron

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_password neutrontest

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
neutron_admin_auth_url http://rhos-keystone-vip:35357/v2.0
```

```
[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
firewall_driver nova.virt.firewall.NoopFirewallDriver

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
libvirt_vif_driver nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
```

12. Configure Nova to use the VNC proxy.

```
[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
vncserver_proxyclient_address $(ip addr show dev br-em2 scope global |
grep inet | sed -e 's#.*inet ##g'  -e  's#/.*##g')

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
vncserver_listen 0.0.0.0

[root@rhos7 ~]# openstack-config --set /etc/nova/nova.conf DEFAULT
novncproxy_base_url http://rhos-nova-vip:6080/vnc_auto.html
```

13. Configure */etc/nova/api-paste.ini* to ensure compatibility across OpenStack and EC2 APIs.

```
[root@rhos7 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken auth_host rhos-keystone-vip

[root@rhos7 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_tenant_name services

[root@rhos7 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_user compute

[root@rhos7 ~]# openstack-config --set /etc/nova/api-paste.ini
filter:authtoken admin_password novatest
```

14. Configure Neutron.

```
[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
auth_strategy keystone

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken auth_host rhos-keystone-vip

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_tenant_name services

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_user neutron

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf
keystone_authtoken admin_password neutrontest

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
rpc_backend neutron.openstack.common.rpc.impl_qpid
```

```
[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
qpid_hostname rhos-qpid-vip

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf database
connection mysql://neutron:neutrontest@rhos-mysql-vip:3306/ovs_neutron

[root@rhos7 ~]# openstack-config --set /etc/neutron/neutron.conf DEFAULT
core_plugin
neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2

[root@rhos7 ~]# ln -s
/etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini
/etc/neutron/plugin.ini
```

15. Configure the Open vSwitch plugin to support VLANs,

```
[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini DATABASE
sql_connection mysql://neutron:neutrontest@rhos-mysql-vip/ovs_neutron

[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
enable_tunneling False

[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
tenant_network_type vlan

[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
integration_bridge br-int

[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
network_vlan_ranges physnet1:1000:1010

[root@rhos7 ~]# openstack-config --set /etc/neutron/plugin.ini OVS
bridge_mappings physnet1:br-p3p1
```

16. Start the cluster.

```
[root@rhos7 ~]# chkconfig pacemaker on

[root@rhos7 ~]# pcs cluster setup --name nova-rhos7 rhos7

[root@rhos7 ~]# pcs cluster start
Starting Cluster...
```

17. Create the fencing device.

```
[root@rhos7 ~]# pcs stonith create rhos7-fence fence_ipmilan params
login="root" passwd="examplepasswd" action="reboot"
ipaddr="10.19.143.161" lanplus="" verbose="" pcmk_host_list=rhos5-priv

[root@rhos7 ~]# pcs stonith show rhos7-fence
Resource: rhos7-fence (class=stonith type=fence_ipmilan)
  Attributes: login=root passwd=examplepasswd action=reboot
ipaddr=10.19.143.161 lanplus= verbose= pcmk_host_list=rhos7-priv
```

```
    Operations: monitor interval=60s (rhos7-fence-monitor-interval-60s)
```

18. Create the cluster resources.

```
[root@rhos7 ~]# pcs resource create openvswitch lsb:openvswitch

[root@rhos7 ~]# pcs resource create neutron-db-check lsb:neutron-db-check
meta failure-timeout=5

[root@rhos7 ~]# pcs resource create neutron-openvswitch-agent
lsb:neutron-openvswitch-agent

[root@rhos7 ~]# pcs resource create messagebus lsb:messagebus

[root@rhos7 ~]# pcs resource create libvirtd lsb:libvirtd

[root@rhos7 ~]# pcs resource create nova-compute lsb:openstack-nova-
compute
```

19. Verify the resource start.

```
[root@rhos7 ~]# pcs status
Cluster name: nova-rhos7
Last updated: Thu Jun 12 12:34:45 2014
Last change: Thu Jun 12 12:34:13 2014 via cibadmin on rhos7
Stack: cman
Current DC: rhos7 - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
1 Nodes configured
7 Resources configured

Online: [ rhos7 ]

Full list of resources:

 rhos7-fence     (stonith:fence_ipmilan):    Started rhos7
 openvswitch     (lsb:openvswitch):    Started rhos7
 neutron-db-check     (lsb:neutron-db-check):     Started rhos7
 neutron-openvswitch-agent  (lsb:neutron-openvswitch-agent):  Started
rhos7
 messagebus(lsb:messagebus):     Started rhos7
 libvirtd  (lsb:libvirtd):  Started rhos7
 nova-compute    (lsb:openstack-nova-compute):     Started rhos7
```

20. Create the cluster resource constraints.

```
[root@rhos7 ~]# pcs constraint order start openvswitch then neutron-db-
check
Adding openvswitch neutron-db-check (kind: Mandatory) (Options: first-
action=start then-action=start)

[root@rhos7 ~]# pcs constraint order start neutron-db-check then neutron-
openvswitch-agent
Adding neutron-db-check neutron-openvswitch-agent (kind: Mandatory)
```

```
(Options: first-action=start then-action=start)

[root@rhos7 ~]# pcs constraint order start neutron-openvswitch-agent then
messagebus
Adding neutron-openvswitch-agent messagebus (kind: Mandatory) (Options:
first-action=start then-action=start)

[root@rhos7 ~]# pcs constraint order start messagebus then libvirtd
Adding messagebus libvirtd (kind: Mandatory) (Options: first-action=start
then-action=start)

[root@rhos7 ~]# pcs constraint order start libvirtd then nova-compute
Adding libvirtd nova-compute (kind: Mandatory) (Options: first-
action=start then-action=start)
```

21. Verify the constraints.

```
[root@rhos7 ~]# pcs constraint show
Location Constraints:
Ordering Constraints:
  start openvswitch then start neutron-db-check
  start neutron-db-check then start neutron-openvswitch-agent
  start neutron-openvswitch-agent then start messagebus
  start messagebus then start libvirtd
  start libvirtd then start nova-compute
Colocation Constraints:
```

22. Create the */root/keystonerc_admin* file on the controller nodes.

```
export OS_USERNAME=admin
export OS_TENANT_NAME=admin
export OS_PASSWORD=keystonetest
export OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
export PS1='[\u@\h \W(keystone_admin)]\$ '
```

23. Run a Nova command from the compute node.

```
[root@rhos7 ~]# source /root/keystonerc_admin

[root@rhos7 ~]# nova usage-list
Usage from 2014-05-15 to 2014-06-13:
+-----------+---------+-------------+-----------+---------------+
| Tenant ID | Servers | RAM MB-Hours | CPU Hours | Disk GB-Hours |
+-----------+---------+-------------+-----------+---------------+
+-----------+---------+-------------+-----------+---------------+
```

24. List the Nova processes.

```
[root@rhos7 ~(keystone_admin)]# ps -ef | grep nova
nova      1777     1 11 12:34 ?        00:00:03 /usr/bin/python
/usr/bin/nova-compute --logfile /var/log/nova/compute.log
root      2131 30878  0 12:34 pts/2    00:00:00 grep nova
```

25. View Nova services from the compute node. All services should be enabled and up. This example shows output after all compute nodes are configured.

```
[root@rhos7 ~(keystone_admin)]# nova service-list
+------------------+-------------------------------------+---------------+
| Binary           | Host                                | Zone     |
Status  | State | Updated_at                | Disabled Reason |
+------------------+-------------------------------------+---------------+
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:53.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:53.000000 | None            |
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:53.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:56.000000 | None            |
| nova-scheduler   | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:56.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:56.000000 | None            |
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-06-12T17:34:56.000000 | None            |
| nova-compute     | rhos5.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-06-12T17:34:55.000000 | None            |
| nova-compute     | rhos6.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-06-12T17:34:53.000000 | None            |
| nova-compute     | rhos7.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-06-12T17:34:56.000000 | None            |
+------------------+-------------------------------------+---------------+
```

26. Check for SELinux errors.

```
[root@rhos7 ~]# ausearch -m avc -i; audit2allow -al
<no matches>
<no matches>
```

4.5 Test Solution

This section describes the steps taken to validate basic OpenStack functionality of this reference architecture. The test case includes:

- Commands to verify the initial configuration
- Adding an image to Glance
- Creating a Keystone tenant and user
- Defining a network that includes public and private switches and a router
- Adding a key pair
- Deploying 2 virtual machines
- Verifying the virtual machines can communicate over the private network
- Creating and attaching a Cinder volume

- Adding a floating IP address to a virtual machine and accessing it from the public network

4.5.1 Verify the Initial Configuration

1. Source */etc/keystonerc_admin*.

```
[root@rhos2 ~]# source /root/keystonerc_admin
```

2. List the Neutron agents. Notice the shared host name defined in **4.3.8 Implement HA Neutron Network Agents** for the L3 agent.

```
[root@rhos2 ~(keystone_admin)]# neutron agent-list
+--------------------------------------+--------------------+
| id                                   | agent_type   | host
| alive | admin_state_up |
+--------------------------------------+-------+-------------+
| 0286f468-b793-4900-a181-db263c93f46b | Open vSwitch agent |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| 07a14e05-d82c-47ee-8783-e8a9ed540799 | Open vSwitch agent |
rhos7.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| 2c9c3f26-1f42-4bd8-bced-2ed64930873e | DHCP agent         |
rhos2.cloud.lab.eng.bos.redhat.com | xxx   | True           |
| 400b3b23-6473-4b68-ab9f-71384e408a21 | Open vSwitch agent |
rhos6.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| 9928fe15-9898-4660-ad3f-836d4889d554 | Open vSwitch agent |
rhos5.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| d1145a2f-df8d-4638-834c-3418249cf7b3 | DHCP agent         |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| e9e3bda3-3455-4db3-810b-19ac72758c18 | Open vSwitch agent |
rhos2.cloud.lab.eng.bos.redhat.com | xxx   | True           |
| eeda618c-d4ab-4ed3-9d05-264862590899 | L3 agent           |
rhos4-neutron-n                    | :-)   | True           |
+--------------------------------------+--------------------+
```

3. List the Nova services.

```
[root@rhos2 ~(keystone_admin)]# nova service-list
+---------+-------+---------------------------+-----------------+
| Binary           | Host                                | Zone     |
Status  | State | Updated_at                | Disabled Reason |
+---------+-------+---------------------------+-----------------+
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:45.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:45.000000 | None            |
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:45.000000 | None            |
| nova-scheduler   | rhos2.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:46.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:47.000000 | None            |
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal |
```

```
enabled | up    | 2014-05-01T15:09:47.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal |
enabled | up    | 2014-05-01T15:09:47.000000 | None            |
| nova-compute     | rhos7.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-05-01T15:09:45.000000 | None            |
| nova-compute     | rhos6.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-05-01T15:09:50.000000 | None            |
| nova-compute     | rhos5.cloud.lab.eng.bos.redhat.com | nova     |
enabled | up    | 2014-05-01T15:09:45.000000 | None            |
+---------+-------+----------------------------+-----------------+
```

4. List the Cinder services.

```
[root@rhos2 ~(keystone_admin)]# cinder service-list
+------------------+-------------------------------------+------+--------+
|      Binary      |                Host                 | Zone |  Status
| State |         Updated_at          |
+------------------+-------------------------------------+------+--------+
| cinder-scheduler | rhos2.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:48.000000 |
| cinder-scheduler | rhos3.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:46.000000 |
| cinder-scheduler | rhos4.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:47.000000 |
|  cinder-volume   | rhos2.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:51.000000 |
|  cinder-volume   | rhos3.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:53.000000 |
|  cinder-volume   | rhos4.cloud.lab.eng.bos.redhat.com | nova | enabled
|   up  | 2014-05-05T15:31:50.000000 |
+------------------+-------------------------------------+------+--------+
```

4.5.2 Prepare the Test Environment

This section describes steps to create a realistic test environment. Execute the commands in this section as the OpenStack administrator. Execute the commands in subsequent sections as a Member user.

1. Source */etc/keystonerc_admin* to execute commands as the OpenStack administrator.

```
[root@rhos7 ~]# source /root/keystonerc_admin

[root@rhos7 ~(keystone_admin)]# env | grep OS_
OS_PASSWORD=keystonetest
OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

2. Create a provider network. The external network *physnet1* is mapped to the *br-p3p1* bridge.

```
[root@rhos7 ~]# neutron net-create ext_net --router:external=True
```

```
--provider:network_type flat --provider:physical_network physnet1
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 3469b7b5-561c-4d31-b2ea-0ed94a0e17fb |
| name                      | ext_net                              |
| provider:network_type     | flat                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 6824420035fd463da599bc91fd5d213a     |
+---------------------------+--------------------------------------+
```

3. Create a subnet named *public*. In this example the *public* subnet mirrors the lab management network. The floating IP address allocation pool is a range of IP addresses on the lab network reserved for OpenStack use.

```
[root@rhos7 ~]# neutron subnet-create --name public --gateway
10.19.143.254 --allocation-pool start=10.19.137.112,end=10.19.137.116
ext_net 10.19.136.0/21 -- --enable_dhcp=False
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "10.19.137.112", "end": "10.19.137.116"} |
| cidr              | 10.19.136.0/21                                       |
| dns_nameservers   |                                                      |
| enable_dhcp       | False                                                |
| gateway_ip        | 10.19.143.254                                        |
| host_routes       |                                                      |
| id                | 2418dd59-723f-4acd-8b06-63f5d6f88dde                 |
| ip_version        | 4                                                    |
| name              | public                                               |
| network_id        | 3469b7b5-561c-4d31-b2ea-0ed94a0e17fb                 |
| tenant_id         | 6824420035fd463da599bc91fd5d213a                     |
+-------------------+------------------------------------------------------+
```

4. Import the virtual machine disk image.

```
[root@rhos7 ~(keystone_admin)]# glance image-create --name rhel-server2
--is-public true --disk-format qcow2 --container-format bare --file
/pub/projects/rhos/common/images/rhel-server-x86_64-kvm-6.4_20130130.0-
4.qcow2

[root@rhos7 ~(keystone_admin)]# glance image-list
+--------------------------------------+--------------+-----------------+
| ID                                   | Name         | Disk Format |
Container Format | Size      | Status |
+--------------------------------------+--------------+-----------------+
```

```
| 6e087c2c-fb8f-4017-b677-18ae3b92118f | RHEL 6.5     | qcow2       |
bare             | 342818816 | active |
| 2b37090a-1ec8-40a3-944a-e55b0f0979bb | rhel-server2 | qcow2       |
bare             | 699592704 | saving |
+--------------------------------------+--------------+-----------------+
```

5. Create a tenant and a tenant user. In this example the user is named *refarch*. The tenant is named *refarch-tenant*.

```
[root@rhos7 ~]# keystone user-create --name refarch --pass refarch
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 262099533f6648248cf7a38468dc736d |
|   name   |             refarch              |
+----------+----------------------------------+

[root@rhos7 ~]# keystone tenant-create --name refarch-tenant
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|   enabled   |               True               |
|      id     | 8958b9efa43549d7887fcb1bc5dadfe4 |
|     name    |          refarch-tenant          |
+-------------+----------------------------------+
```

6. Add the Member role to the tenant user.

```
[root@rhos7 ~]# keystone user-role-add --user-id refarch --tenant-id
refarch-tenant --role-id _member_

[root@rhos7 ~]# keystone user-role-list --user-id refarch --tenant-id
refarch-tenant
+----------------------------------+----------------------------------+
|                id                |   name   |             user_id
|            tenant_id             |
+----------------------------------+----------------------------------+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
262099533f6648248cf7a38468dc736d | 8958b9efa43549d7887fcb1bc5dadfe4 |
+----------------------------------+----------------------------------+
```

7. Create a file named */root/keystonerc_refarch* that contains environment variables for the tenant user.

```
export OS_USERNAME=refarch
export OS_TENANT_NAME=refarch-tenant
export OS_PASSWORD=refarch
export OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
export PS1='[\u@\h \W(refarch_member)]\$ '
```

4.5.3 Create the Tenant Network

Execute the commands in this section as the tenant user. These commands create the tenant network, subnet, and router. They also add rules to the tenant security group.

1. Source */root/keystonerc_refarch* to execute commands as the tenant user.

```
[root@rhos7 ~(keystone_admin)]# source /root/keystonerc_refarch

[root@rhos7 ~(refarch_member)]# env | grep OS_
OS_PASSWORD=refarch
OS_AUTH_URL=http://rhos-keystone-vip:35357/v2.0/
OS_USERNAME=refarch
OS_TENANT_NAME=refarch-tenant
```

2. Add SSH and ICMP to the default security group for the tenant.

```
[root@rhos7 ~(refarch_member)]# neutron security-group-rule-create
--direction ingress --protocol icmp b31f61cd-f102-43e1-a13f-6b3224a639ca
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | f8577737-9654-4687-95c7-3e83f8ed0a25 |
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          | icmp                                 |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b31f61cd-f102-43e1-a13f-6b3224a639ca |
| tenant_id         | 8958b9efa43549d7887fcb1bc5dadfe4     |
+-------------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron security-group-rule-create
--direction ingress--protocol tcp --port_range_min 22 --port_range_max 22
b31f61cd-f102-43e1-a13f-6b3224a639ca
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 649400b9-6740-44c9-a88a-85eed6dd9da1 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b31f61cd-f102-43e1-a13f-6b3224a639ca |
| tenant_id         | 8958b9efa43549d7887fcb1bc5dadfe4     |
+-------------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron security-group-rule-create
```

```
--direction ingress --protocol tcp --port_range_min 80 --port_range_max
80 b31f61cd-f102-43e1-a13f-6b3224a639ca
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 0734119c-7b67-4ecb-99ee-3b265c508843 |
| port_range_max    | 80                                   |
| port_range_min    | 80                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b31f61cd-f102-43e1-a13f-6b3224a639ca |
| tenant_id         | 8958b9efa43549d7887fcb1bc5dadfe4     |
+-------------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron security-group-show b31f61cd-
f102-43e1-a13f-6b3224a639ca
+-----------------------------------------------------------------------------+
| Field                | Value
| description          | default
| id                   | b31f61cd-f102-43e1-a13f-6b3224a639ca
| name                 | default
| security_group_rules | {"remote_group_id": null, "direction":
"ingress", "remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"8958b9efa43549d7887fcb1bc5dadfe4", "port_range_max": 80,
"security_group_id": "b31f61cd-f102-43e1-a13f-6b3224a639ca",
"port_range_min": 80, "ethertype": "IPv4", "id": "0734119c-7b67-4ecb-
99ee-3b265c508843"}                                          |
|                      | {"remote_group_id": "b31f61cd-f102-43e1-a13f-
6b3224a639ca", "direction": "ingress", "remote_ip_prefix": null,
"protocol": null, "tenant_id": "8958b9efa43549d7887fcb1bc5dadfe4",
"port_range_max": null, "security_group_id": "b31f61cd-f102-43e1-a13f-
6b3224a639ca", "port_range_min": null, "ethertype": "IPv4", "id":
"3b4acb36-07b7-484d-aee7-622b290fa667"} |
|                      | {"remote_group_id": null, "direction":
"ingress", "remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"8958b9efa43549d7887fcb1bc5dadfe4", "port_range_max": 22,
"security_group_id": "b31f61cd-f102-43e1-a13f-6b3224a639ca",
"port_range_min": 22, "ethertype": "IPv4", "id": "649400b9-6740-44c9-
a88a-85eed6dd9da1"}                                          |
|                      | {"remote_group_id": "b31f61cd-f102-43e1-a13f-
6b3224a639ca", "direction": "ingress", "remote_ip_prefix": null,
"protocol": null, "tenant_id": "8958b9efa43549d7887fcb1bc5dadfe4",
"port_range_max": null, "security_group_id": "b31f61cd-f102-43e1-a13f-
6b3224a639ca", "port_range_min": null, "ethertype": "IPv6", "id":
"a275b40d-2cc6-452d-87f6-d4fa84d3ed5f"} |
|                      | {"remote_group_id": null, "direction": "egress",
"remote_ip_prefix": null, "protocol": null, "tenant_id":
"8958b9efa43549d7887fcb1bc5dadfe4", "port_range_max": null,
"security_group_id": "b31f61cd-f102-43e1-a13f-6b3224a639ca",
"port_range_min": null, "ethertype": "IPv6", "id": "bb24f88f-845d-43a1-
bab3-546a634091d2"}                                          |
```

```
|                      | {"remote_group_id": null, "direction": "egress",
"remote_ip_prefix": null, "protocol": null, "tenant_id":
"8958b9efa43549d7887fcb1bc5dadfe4", "port_range_max": null,
"security_group_id": "b31f61cd-f102-43e1-a13f-6b3224a639ca",
"port_range_min": null, "ethertype": "IPv4", "id": "ecbc9a11-50fb-4819-
85cd-677921a7ae13"}                                          |
|                      | {"remote_group_id": null, "direction":
"ingress", "remote_ip_prefix": null, "protocol": "icmp", "tenant_id":
"8958b9efa43549d7887fcb1bc5dadfe4", "port_range_max": null,
"security_group_id": "b31f61cd-f102-43e1-a13f-6b3224a639ca",
"port_range_min": null, "ethertype": "IPv4", "id": "f8577737-9654-4687-
95c7-3e83f8ed0a25"}                                          |
| tenant_id            | 8958b9efa43549d7887fcb1bc5dadfe4
+----------------------------------------------------------------------+
```

3. Create a tenant network and a private subnet.

```
[root@rhos7 ~(refarch_member)]# neutron net-create net1
Created a new network:
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| admin_state_up | True                                 |
| id             | 23511d43-ab4a-40bf-b50e-1c90732de3f4 |
| name           | net1                                 |
| shared         | False                                |
| status         | ACTIVE                               |
| subnets        |                                      |
| tenant_id      | 8958b9efa43549d7887fcb1bc5dadfe4     |
+----------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron subnet-create --name private net1
172.16.3.0/24 --dns_nameservers list=true 10.19.143.247
Created a new subnet:
+------------------+------------------------------------------------------+
| Field            | Value                                                |
+------------------+------------------------------------------------------+
| allocation_pools | {"start": "172.16.3.2", "end": "172.16.3.254"}       |
| cidr             | 172.16.3.0/24                                        |
| dns_nameservers  | 10.19.143.247                                        |
| enable_dhcp      | True                                                 |
| gateway_ip       | 172.16.3.1                                           |
| host_routes      |                                                      |
| id               | 0e002594-b43d-431b-853f-6d0b01783aef                 |
| ip_version       | 4                                                    |
| name             | private                                              |
| network_id       | 23511d43-ab4a-40bf-b50e-1c90732de3f4                 |
| tenant_id        | 8958b9efa43549d7887fcb1bc5dadfe4                     |
+------------------+------------------------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron net-list
+--------------------------------------+---------+---------------+------+
| id                                   | name    | subnets
+--------------------------------------+---------+---------------+------+
| 23511d43-ab4a-40bf-b50e-1c90732de3f4 | net1    | 0e002594-b43d-431b-
```

```
853f-6d0b01783aef 172.16.3.0/24 |
| 3469b7b5-561c-4d31-b2ea-0ed94a0e17fb | ext_net | 2418dd59-723f-4acd-
8b06-63f5d6f88dde               |
+--------------------------------------+---------+--------------+------+

[root@rhos7 ~(refarch_member)]# neutron subnet-list
+--------------------------------------+---------+--------------+------+
| id                                   | name    | cidr         |
allocation_pools                              |
+--------------------------------------+---------+--------------+------+
| 0e002594-b43d-431b-853f-6d0b01783aef | private | 172.16.3.0/24 |
{"start": "172.16.3.2", "end": "172.16.3.254"} |
+--------------------------------------+---------+--------------+------+
```

4. Create a router.

```
[root@rhos7 ~(refarch_member)]# neutron router-create route1
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 8431405c-8e27-434f-b7e4-22cafcbb6ad4 |
| name                  | route1                               |
| status                | ACTIVE                               |
| tenant_id             | 8958b9efa43549d7887fcb1bc5dadfe4     |
+-----------------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# neutron router-show route1
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 8431405c-8e27-434f-b7e4-22cafcbb6ad4 |
| name                  | route1                               |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | 8958b9efa43549d7887fcb1bc5dadfe4     |
+-----------------------+--------------------------------------+
```

5. Connect the private tenant subnet to the router.

```
[root@rhos7 ~(refarch_member)]# subnet_id=$(neutron subnet-list | awk ' /
172.16.3.0/ { print $2 } ')

[root@rhos7 ~(refarch_admin)]# neutron router-interface-add route1
$subnet_id
Added interface 8e5635b6-6d40-4417-9c20-cf8d0987be8c to router route1.
```

4.5.4 Add the External Network to the Tenant Router

Add the external network defined in **4.5.2 Prepare the Test Environment** to the tenant

router. This allows communication between the public and private subnets.

1. Find the router Keystone ID.

```
[root@rhos7 ~(refarch_admin)]# route_id=$(neutron router-list | awk '
/route1/ { print $2 } ')
```

2. Find the external network Keystone ID.

```
[root@rhos7 ~(refarch_member)]# ext_net_id=$(neutron net-list | awk '
/ext_net/ { print $2 } ')
```

3. Set the router gateway to the external network ID.

```
[root@rhos7 ~(refarch_member)]# neutron router-gateway-set $route_id
$ext_net_id
Set gateway for router 8431405c-8e27-434f-b7e4-22cafcbb6ad4
```

4. Verify the router gateway.

```
[root@rhos7 ~(refarch_member)]# neutron router-show $route_id
+-----------------------+-----------------------------------------------+
| Field                 | Value
+-----------------------+-----------------------------------------------+
| admin_state_up        | True
| external_gateway_info | {"network_id": "3469b7b5-561c-4d31-b2ea-
0ed94a0e17fb", "enable_snat": true} |
| id                    | 8431405c-8e27-434f-b7e4-22cafcbb6ad4
| name                  | route1
| routes                |
| status                | ACTIVE
| tenant_id             | 8958b9efa43549d7887fcb1bc5dadfe4
+-----------------------+-----------------------------------------------+
```

5. View the gateway port on the router to find the gateway IP address.

```
[root@rhos7 ~(refarch_member)]# neutron port-list --device-id 8431405c-
8e27-434f-b7e4-22cafcbb6ad4
+-----------------------+-----------------------------------------------+
| id                                    | name | mac_address       |
fixed_ips
+-----------------------+-----------------------------------------------+
| 8e5635b6-6d40-4417-9c20-cf8d0987be8c |      | fa:16:3e:97:c6:f1 |
{"subnet_id": "0e002594-b43d-431b-853f-6d0b01783aef", "ip_address":
"172.16.3.1"} |
+-----------------------+-----------------------------------------------+
```

4.5.5 Boot the Virtual Machine Instances

Create a key pair and boot two virtual machine instances as the tenant user.

1. Create a key pair and set its permissions.

```
[root@rhos7 ~(refarch_member)]# nova keypair-add refarchkp >
/root/refarchkp.pem
```

```
[root@rhos7 ~(refarch_member)]# chmod 600 /root/refarchkp.pem
```

2. Boot two instances.

```
[root@rhos7 ~(refarch_member)]# nova boot --flavor 2 --image rhel-server2
--key-name refarchkp inst1
+--------------------------------------+--------------------------------+
| Property                             | Value
+--------------------------------------+--------------------------------+
| OS-EXT-STS:task_state                | scheduling
| image                                | rhel-server2
| OS-EXT-STS:vm_state                  | building
| OS-SRV-USG:launched_at               | None
| flavor                               | m1.small
| id                                   | 515e2679-25ae-4975-b662-
d959d1ba4412 |
| security_groups                      | [{u'name': u'default'}]
| user_id                              | 262099533f6648248cf7a38468dc736d
| OS-DCF:diskConfig                    | MANUAL
| accessIPv4                           |
| accessIPv6                           |
| progress                             | 0
| OS-EXT-STS:power_state               | 0
| OS-EXT-AZ:availability_zone          | nova
| config_drive                         |
| status                               | BUILD
| updated                              | 2014-06-12T17:38:52Z
| hostId                               |
| OS-SRV-USG:terminated_at             | None
| key_name                             | refarchkp
| name                                 | inst1
| adminPass                            | c9BE5pyTe6bJ
| tenant_id                            | 8958b9efa43549d7887fcb1bc5dadfe4
| created                              | 2014-06-12T17:38:51Z
| os-extended-volumes:volumes_attached | []
| metadata                             | {}
+--------------------------------------+--------------------------------+

[root@rhos7 ~(refarch_member)]# nova boot --flavor 2 --image rhel-server2
--key-name refarchkp inst2
+--------------------------------------+--------------------------------+
| Property                             | Value
+--------------------------------------+--------------------------------+
| OS-EXT-STS:task_state                | scheduling
| image                                | rhel-server2
| OS-EXT-STS:vm_state                  | building
| OS-SRV-USG:launched_at               | None
| flavor                               | m1.small
| id                                   | 7d37a28d-355e-40ef-833a-
698e555daa05 |
| security_groups                      | [{u'name': u'default'}]
| user_id                              | 262099533f6648248cf7a38468dc736d
| OS-DCF:diskConfig                    | MANUAL
```

```
| accessIPv4                           |
| accessIPv6                           |
| progress                             | 0
| OS-EXT-STS:power_state               | 0
| OS-EXT-AZ:availability_zone          | nova
| config_drive                         |
| status                               | BUILD
| updated                              | 2014-06-12T17:38:54Z
| hostId                               |
| OS-SRV-USG:terminated_at             | None
| key_name                             | refarchkp
| name                                 | inst2
| adminPass                            | Xtep2BaJ7eWS
| tenant_id                            | 8958b9efa43549d7887fcb1bc5dadfe4
| created                              | 2014-06-12T17:38:54Z
| os-extended-volumes:volumes_attached | []
| metadata                             | {}
+--------------------------------------+--------------------------------+
```

3. View the instances.

```
[root@rhos7 ~(refarch_member)]# nova list
+--------------------------------------+--------------------------------+
| ID                                   | Name  | Status | Task State |
Power State | Networks                       |
+--------------------------------------+--------------------------------+
| 515e2679-25ae-4975-b662-d959d1ba4412 | inst1 | ACTIVE | None       |
Running     | net1=172.16.3.2 |
| 7d37a28d-355e-40ef-833a-698e555daa05 | inst2 | ACTIVE | None       |
Running     | net1=172.16.3.4                |
+--------------------------------------+--------------------------------+
```

4.5.6 Associate a Floating IP Address with an Instance

Create a floating IP address and associate it with an instance. Ping the instance via the floating IP address.

1. Create a floating IP address from the public subnet pool. Save its Keystone ID to an environment variable.

```
[root@rhos7 ~(refarch_member)]# floatip_id=$(neutron floatingip-create
ext_net | awk ' / id/ { print $4 } ')

[root@rhos7 ~(refarch_member)]# neutron floatingip-list
+--------------------------------------+---------------------+---------+
| id                                   | fixed_ip_address |
floating_ip_address | port_id |
+--------------------------------------+---------------------+---------+
| a8566862-2b03-4b39-96b0-46b16ed4fb71 |                     | 10.19.137.113
+--------------------------------------+---------------------+---------+
```

2. Find the port ID of an instance.

```
[root@rhos7 ~(refarch_member)]# inst1_id=$(nova list | awk ' /inst1/
{ print $2 } ')
```

```
[root@rhos7 ~(refarch_member)]# inst1_port=$(neutron port-list
--device_id $inst1_id | awk ' /ip_address/ { print $2 } ')
```

3. Associate the floating IP address with the port.

```
[root@rhos7 ~(refarch_member)]# neutron floatingip-associate $floatip_id
$inst1_port
Associated floatingip a8566862-2b03-4b39-96b0-46b16ed4fb71
```

4. View the association.

```
[root@rhos7 ~(refarch_member)]# neutron floatingip-list
+--------------------------------------+------------------+-------------+
| id                                   | fixed_ip_address |
floating_ip_address | port_id                              |
+--------------------------------------+------------------+-------------+
| a8566862-2b03-4b39-96b0-46b16ed4fb71 | 172.16.3.2       | 10.19.137.113
| b514d5a9-4be1-40c1-bd3d-b7afca856771 |
+--------------------------------------+------------------+-------------+
```

5. Ping the first instance via floating IP address.

```
[root@rhos7 ~(refarch_member)]# float_ip=$(neutron floatingip-list | grep
$floatip_id | awk ' { print $6 } ')

[root@rhos7 ~(refarch_member)]# echo $float_ip
10.19.137.113

[root@rhos7 ~(refarch_member)]# ping -c 3 $float_ip
PING 10.19.137.113 (10.19.137.113) 56(84) bytes of data.
64 bytes from 10.19.137.113: icmp_seq=1 ttl=63 time=3.60 ms
64 bytes from 10.19.137.113: icmp_seq=2 ttl=63 time=0.659 ms
64 bytes from 10.19.137.113: icmp_seq=3 ttl=63 time=0.698 ms
```

6. Connect to the first instance via SSH.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem $float_ip uptime
 15:18:29 up 21:38,  0 users,  load average: 0.00, 0.00, 0.00
```

7. Find the IP address of the second instance.

```
[root@rhos7 ~(refarch_member)]# inst2_ip=$(nova show inst2 | awk ' /net1/
{ print $5 } ' | cut -f1 -d,)

[root@rhos7 ~(refarch_member)]# echo -e "inst2_ip: $inst2_ip"
inst2_ip: 172.16.3.4
```

8. Ping the second instance from the first. This verifies that both instances are online and that the tenant network is correctly configured.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem $float_ip ping -c 3
```

```
$inst2_ip
PING 172.16.3.4 (172.16.3.4) 56(84) bytes of data.
64 bytes from 172.16.3.4: icmp_seq=1 ttl=64 time=3.09 ms
64 bytes from 172.16.3.4: icmp_seq=2 ttl=64 time=0.709 ms
64 bytes from 172.16.3.4: icmp_seq=3 ttl=64 time=0.716 ms

--- 172.16.3.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.709/1.507/3.098/1.125 ms
```

4.5.7 Attach a Persistent Storage Volume to an Instance

Create a Cinder volume. Attach it to a virtual machine instance.

1. Create a Cinder volume.

```
[root@rhos7 ~(refarch_member)]# cinder create --display-name test 1
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2014-06-12T17:46:07.700141      |
| display_description |                 None                 |
|     display_name    |                 test                 |
|          id         | b74038fc-5129-4d66-80d1-37d97aadc617 |
|       metadata      |                  {}                  |
|         size        |                  1                   |
|     snapshot_id     |                 None                 |
|     source_volid    |                 None                 |
|        status       |              creating                |
|     volume_type     |                 None                 |
+---------------------+--------------------------------------+

[root@rhos7 ~(refarch_member)]# cinder show b74038fc-5129-4d66-80d1-
37d97aadc617
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
|     attachments     |                  []                  |
|  availability_zone  |                 nova                 |
|       bootable      |                false                 |
|      created_at     |      2014-06-12T17:46:07.000000      |
| display_description |                 None                 |
|     display_name    |                 test                 |
|          id         | b74038fc-5129-4d66-80d1-37d97aadc617 |
|       metadata      |                  {}                  |
|         size        |                  1                   |
|     snapshot_id     |                 None                 |
|     source_volid    |                 None                 |
|        status       |              available               |
|     volume_type     |                 None                 |
+---------------------+--------------------------------------+
```

2. Find the volume's ID.

```
[root@rhos7 ~(refarch_member)]# volid=$(nova volume-list | awk ' /test/ { print $2 } ')

[root@rhos7 ~(refarch_member)]# echo -e "volid = $volid"
volid = b74038fc-5129-4d66-80d1-37d97aadc617
```

3. Attach the volume to an instance. Save the volume name to an environment variable.

```
[root@rhos7 ~(refarch_member)]# volname=$(nova volume-attach inst1 $volid auto | awk ' /device/ { print $4 } ')

[root@rhos7 ~(refarch_member)]# echo -e "$volname"
/dev/vdb
```

4. Verify the volume attached to the instance.

```
[root@rhos7 ~(refarch_member)]# cinder list
+--------------------------------------+--------+--------------+------+-+
|                  ID                  | Status | Display Name | Size |
Volume Type | Bootable |             Attached to              |
+--------------------------------------+--------+--------------+------+-+
| b74038fc-5129-4d66-80d1-37d97aadc617 | in-use |     test     |  1   |
None    |  false   | 515e2679-25ae-4975-b662-d959d1ba4412 |
+--------------------------------------+--------+--------------+------+-+
```

5. Verify the instance detects the volume.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem $float_ip partprobe -s
Warning: WARNING: the kernel failed to re-read the partition table on
/dev/vda (Device or resource busy).  As a result, it may not reflect all
of your changes until after reboot.

[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 grep vdb /proc/partitions
252       16    1048576 vdb
```

6. Configure a partition on the volume.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 parted /dev/vdb mklabel gpt
Information: You may need to update /etc/fstab.
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 mkfs.ext4 /dev/vdb
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
65536 inodes, 262144 blocks
13107 blocks (5.00%) reserved for the super user
```

```
First data block=0
Maximum filesystem blocks=268435456
8 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 32 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

7. Create a mount point and mount the file system.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 mkdir
-p /mnt/vol

[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113
mount /dev/vdb /mnt/vol

[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 mount
/dev/sda1 on / type ext4 (rw,noatime)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs
(rw,rootcontext="system_u:object_r:tmpfs_t:s0")
none on /dev/shm type tmpfs
(rw,rootcontext="system_u:object_r:tmpfs_t:s0")
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/vdb on /mnt/vol type ext4 (rw)
```

8. Create a test file on the file system.

```
[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113
touch /mnt/vol/testfile

[root@rhos7 ~(refarch_member)]# ssh -i refarchkp.pem 10.19.137.113 ls -al
/mnt/vol
total 24
drwxr-xr-x. 3 root root  4096 Jun 12 17:48 .
drwxr-xr-x. 3 root root  4096 Jun 12 17:48 ..
drwx------. 2 root root 16384 Jun 12 17:48 lost+found
-rw-r--r--. 1 root root     0 Jun 12 17:48 testfile
```

4.5.8 Test Failover

This section tests failover by disabling Pacemaker on the service owning controller.

1. Identify which controller started the active/passice Neutron L3 agent. In this example it

is **rhos4**.

```
[root@rhos4 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 13:04:35 2014
Last change: Thu Jun 12 12:26:56 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
3 Nodes configured
61 Resources configured

Online: [ rhos2-priv rhos3-priv rhos4-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos4-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm      (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
 Resource Group: neutron-agents-pre
     neutron-db-check-pre   (lsb:neutron-db-check):     Started rhos4-priv
     neutron-ovs-cleanup    (lsb:neutron-ovs-cleanup):  Started rhos4-priv
     neutron-netns-cleanup  (lsb:neutron-netns-cleanup):Started rhos4-priv
     neutron-agent-watch    (lsb:neutron-agent-watch):  Started rhos4-priv
 neutron-openvswitch-agent  (lsb:neutron-openvswitch-agent):  Started rhos4-
priv
 neutron-dhcp-agent    (lsb:neutron-dhcp-agent):   Started rhos4-priv
 neutron-l3-agent      (lsb:neutron-l3-agent):     Started rhos4-priv
 neutron-metadata-agent     (lsb:neutron-metadata-agent):    Started rhos4-
priv
```

```
 Clone Set: nova-consoleauth-clone [nova-consoleauth]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-novncproxy-clone [nova-novncproxy]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: nova-api-clone [nova-api]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 nova-scheduler  (lsb:openstack-nova-scheduler):   Started rhos3-priv
 Clone Set: nova-conductor-clone [nova-conductor]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
 Clone Set: httpd-clone [httpd]
     Started: [ rhos2-priv rhos3-priv rhos4-priv ]
```

2. As the Keystone admin, verify the Neutron agents and Nova services are up. In this example Open vSwitch and DHCP agents are running on **rhos4**. The L3 agent is running on the common host name *rhos4-neutron-n*.

```
[root@rhos4 ~]# source /root/keystonerc_admin

[root@rhos4 ~(keystone_admin)]# neutron agent-list
+--------------------------------------+--------------------+--------------+
| id                                   | agent_type         | host
| alive | admin_state_up |
+--------------------------------------+--------------------+--------------+
| 19fa0c68-e824-496f-9c0e-95349e57a767 | Open vSwitch agent |
rhos6.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| 9c2d7ec3-1dd5-470a-aac8-8d59ae4d70a3 | Open vSwitch agent |
rhos7.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| bca7041b-67de-4e72-8cdf-e1ef261ea191 | DHCP agent         |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| c1dd62ca-a745-4774-aae3-ede0acb7b7a1 | L3 agent           | rhos4-neutron-
n                    | :-)   | True           |
| e2caf702-9d7b-4d90-8ce8-2bd671138efc | Open vSwitch agent |
rhos5.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| fe3db77c-e63b-4852-83e7-315135412adf | Open vSwitch agent |
rhos4.cloud.lab.eng.bos.redhat.com | :-)   | True           |
+--------------------------------------+--------------------+--------------+

[root@rhos4 ~(keystone_admin)]# nova service-list
+------------------+-----------------------------------+----------+-------+
| Binary           | Host                                | Zone     | Status
| State | Updated_at                | Disabled Reason |
+------------------+-----------------------------------+----------+-------+
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:43.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:43.000000 | None            |
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:43.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:46.000000 | None            |
| nova-scheduler   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:46.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:46.000000 | None            |
```

```
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:04:46.000000 | None            |
| nova-compute     | rhos5.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:04:45.000000 | None            |
| nova-compute     | rhos6.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:04:43.000000 | None            |
| nova-compute     | rhos7.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:04:46.000000 | None            |
+------------------+------------------------------------+----------+-------+
```

3. The **`ip netns`** command reveals that the L3 agent is running on *rhos4*.

```
[root@rhos4 ~(keystone_admin)]# ip netns
qdhcp-23511d43-ab4a-40bf-b50e-1c90732de3f4
qrouter-8431405c-8e27-434f-b7e4-22cafcbb6ad4
```

4. Stop Pacemaker on the L3 agent-owning controller. This stops all HA services started on the controller node.

```
[root@rhos4 ~(keystone_admin)]# service pacemaker stop
Waiting for shutdown of managed resources.....              [  OK  ]
Signaling Pacemaker Cluster Manager to terminate            [  OK  ]
Waiting for cluster services to unload.                     [  OK  ]
Stopping cluster:
   Leaving fence domain...                                  [  OK  ]
   Stopping gfs_controld...                                 [  OK  ]
   Stopping dlm_controld...                                 [  OK  ]
   Stopping fenced...                                       [  OK  ]
   Stopping cman...                                         [  OK  ]
   Waiting for corosync to shutdown:                        [  OK  ]
   Unloading kernel modules...                              [  OK  ]
   Unmounting configfs...                                   [  OK  ]
```

5. Verify that the Neutron agents and Nova services that were running on the controller are now stopped. Note that the L3 agent is still up.

```
[root@rhos4 ~(keystone_admin)]# neutron agent-list
+------------------+------------------------------------+----------+-------+
| id                                   | agent_type         | host
| alive | admin_state_up |
+------------------+------------------------------------+----------+-------+
| 19fa0c68-e824-496f-9c0e-95349e57a767 | Open vSwitch agent |
rhos6.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| 9c2d7ec3-1dd5-470a-aac8-8d59ae4d70a3 | Open vSwitch agent |
rhos7.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| bca7041b-67de-4e72-8cdf-e1ef261ea191 | DHCP agent         |
rhos4.cloud.lab.eng.bos.redhat.com | xxx   | True           |
| c1dd62ca-a745-4774-aae3-ede0acb7b7a1 | L3 agent           | rhos4-neutron-
n                    | :-)   | True           |
| ca7b7c1c-06ec-459e-b5fb-718b175728ba | Open vSwitch agent |
rhos3.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| d4c6c101-8d7a-4484-9371-342136b5da44 | DHCP agent         |
rhos3.cloud.lab.eng.bos.redhat.com | :-)   | True           |
```

```
| e2caf702-9d7b-4d90-8ce8-2bd671138efc | Open vSwitch agent |
rhos5.cloud.lab.eng.bos.redhat.com | :-)   | True           |
| fe3db77c-e63b-4852-83e7-315135412adf | Open vSwitch agent |
rhos4.cloud.lab.eng.bos.redhat.com | xxx   | True           |
+------------------+--------------------------------------+----------+-------+

[root@rhos4 ~(keystone_admin)]# nova service-list
+------------------+--------------------------------------+----------+-------+
| Binary           | Host                                 | Zone     | Status
| State | Updated_at                 | Disabled Reason |
+------------------+--------------------------------------+----------+-------+
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| down  | 2014-06-12T18:05:03.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:08:03.000000 | None            |
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:08:03.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| down  | 2014-06-12T18:05:06.000000 | None            |
| nova-scheduler   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:07:56.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:07:56.000000 | None            |
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-12T18:07:56.000000 | None            |
| nova-compute     | rhos5.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:08:07.000000 | None            |
| nova-compute     | rhos6.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:08:07.000000 | None            |
| nova-compute     | rhos7.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-12T18:08:07.000000 | None            |
+------------------+--------------------------------------+----------+-------+
```

6. Re-running the **`ip netns`** command on *rhos4* reveals that this controller no longer owns the namespace.

```
[root@rhos4 ~(keystone_admin)]# ip netns
```

7. Connect to another controller node and check cluster status. It should show that the disabled controller is offline and that the Neutron L3 agent has started elsewhere. In this example the L3 agent is running on *rhos3*.

```
[root@rhos4 ~(keystone_admin)]# ssh -l root rhos3
Last login: Wed Jun 11 16:18:29 2014 from se-
users.cloud.lab.eng.bos.redhat.com
Kickstarted on 2014-06-11

[root@rhos3 ~]# pcs status
Cluster name: osp4-controller
Last updated: Thu Jun 12 13:06:21 2014
Last change: Thu Jun 12 12:26:56 2014 via cibadmin on rhos2-priv
Stack: cman
Current DC: rhos3-priv - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
```

```
3 Nodes configured
61 Resources configured

Node rhos4-priv: OFFLINE (standby)
Online: [ rhos2-priv rhos3-priv ]

Full list of resources:

 rhos2-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 rhos3-fence     (stonith:fence_ipmilan):    Started rhos3-priv
 rhos4-fence     (stonith:fence_ipmilan):    Started rhos2-priv
 Clone Set: qpidd-clone [qpidd]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Resource Group: mysql-group
     mysql-ha-lvm      (ocf::heartbeat:LVM):  Started rhos2-priv
     mysql-fs    (ocf::heartbeat:Filesystem):Started rhos2-priv
     mysql-db    (ocf::heartbeat:mysql):     Started rhos2-priv
 Clone Set: keystone-clone [keystone]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: memcached-clone [memcached]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: glance-fs-clone [glance-fs]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: glance-registry-clone [glance-registry]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: glance-api-clone [glance-api]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: cinder-api-clone [cinder-api]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: cinder-scheduler-clone [cinder-scheduler]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: cinder-volume-clone [cinder-volume]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: neutron-db-check-clone [neutron-db-check]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 neutron-server  (lsb:neutron-server):  Started rhos3-priv
 Resource Group: neutron-agents-pre
     neutron-db-check-pre   (lsb:neutron-db-check):     Started rhos3-priv
     neutron-ovs-cleanup    (lsb:neutron-ovs-cleanup):  Started rhos3-priv
     neutron-netns-cleanup  (lsb:neutron-netns-cleanup):Started rhos3-priv
     neutron-agent-watch    (lsb:neutron-agent-watch):  Started rhos3-priv
 neutron-openvswitch-agent  (lsb:neutron-openvswitch-agent):  Started rhos3-
priv
 neutron-dhcp-agent    (lsb:neutron-dhcp-agent):   Started rhos3-priv
 neutron-l3-agent      (lsb:neutron-l3-agent):     Started rhos3-priv
```

```
 neutron-metadata-agent     (lsb:neutron-metadata-agent):     Started rhos3-
priv
 Clone Set: nova-consoleauth-clone [nova-consoleauth]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: nova-novncproxy-clone [nova-novncproxy]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: nova-api-clone [nova-api]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 nova-scheduler  (lsb:openstack-nova-scheduler):   Started rhos3-priv
 Clone Set: nova-conductor-clone [nova-conductor]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
 Clone Set: httpd-clone [httpd]
     Started: [ rhos2-priv rhos3-priv ]
     Stopped: [ rhos4-priv ]
```

8. Verify the network namespace is running on the new controller node.

```
[root@rhos3 ~]# ip netns
qrouter-8431405c-8e27-434f-b7e4-22cafcbb6ad4
qdhcp-23511d43-ab4a-40bf-b50e-1c90732de3f4
```

9. Copy the tenant key from the compute node to the local controller.

```
[root@rhos3 ~]# scp rhos7:/root/refarchkp.pem .
root@rhos7's password: ********
refarchkp.pem
100% 1676     1.6KB/s   00:00
```

10. Connect to an instance via the floating IP address. Verify that the Cinder volume is still mounted and accessible. This step validates that the agents and services are still operational despite the simulated failure of the active controller.

```
[root@rhos3 ~]# ssh -i /root/refarchkp.pem 10.19.137.113 ls /mnt/vol
lost+found
testfile
```

11. Restart Pacemaker on **rhos4**.

```
[root@rhos4 ~(keystone_admin)]# service pacemaker start
Starting cluster:
   Checking if cluster has been disabled at boot...        [  OK  ]
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
```

```
   Tuning DLM kernel config...                                [  OK  ]
   Starting gfs_controld...                                   [  OK  ]
   Unfencing self...                                          [  OK  ]
   Joining fence domain...                                    [  OK  ]
Starting Pacemaker Cluster Manager                            [  OK  ]
```

12. Verify that the Nova services are back online.

```
[root@rhos4 ~(keystone_admin)]# nova service-list
+------------------+-----------------------------------+----------+-------+
| Binary           | Host                              | Zone     | Status
| State | Updated_at                 | Disabled Reason |
+------------------+-----------------------------------+----------+-------+
| nova-consoleauth | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:08.000000 | None            |
| nova-consoleauth | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:11.000000 | None            |
| nova-consoleauth | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:11.000000 | None            |
| nova-conductor   | rhos4.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:04.000000 | None            |
| nova-scheduler   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:05.000000 | None            |
| nova-conductor   | rhos3.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:13.000000 | None            |
| nova-conductor   | rhos2.cloud.lab.eng.bos.redhat.com | internal | enabled
| up    | 2014-06-13T15:59:13.000000 | None            |
| nova-compute     | rhos5.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-13T15:59:10.000000 | None            |
| nova-compute     | rhos6.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-13T15:59:06.000000 | None            |
| nova-compute     | rhos7.cloud.lab.eng.bos.redhat.com | nova     | enabled
| up    | 2014-06-13T15:59:13.000000 | None            |
+------------------+-----------------------------------+----------+-------+
```

5 Conclusion

This reference architecture describes Red Hat's approach to high availability and scaling with RHEL OSP 4. The approach leverages the enterprise-tested RHEL HA Add-on along with the stable code base afforded by RHEL OSP 4 so that customers can deploy a scale out solution with confidence. The steps described in this reference architecture were performed on realistic hardware in the Systems Engineering lab using only released code. The steps were described without using an installer in order to expose the underlying architecture and design decisions often masked by an installer.

The general approach is that all OpenStack services are managed and monitored in isolated RHEL-HA clusters. A high availability network load balancer provides access to the high availability service endpoints. There are no direct connections from the clients to the services. This approach allows individual services can be scaled as needed. This approach also enables a future vision of scalability where in which every service runs on dedicated servers and scales as needed.

Although OpenStack supports many messaging, database, and load balancer technologies, this document focuses on a solution comprised entirely from shipping Red Hat software components. Red Hat tests and supports all of these components. These components are bundled into a single solution to ensure that Red Hat supports and stands behind the end-to-end stack. Similarly, the HA MySQL database resides in shared storage, although this is not a typical scale-out approach, because Red Hat both tests and supports this configuration in many production deployments.

Appendix A: Revision History

Revision 1.0	Wednesday June 18, 2014	Jacob Liberman

- Incorporated engineering review
- First draft

Revision 0.4	Thursday June 12, 2014	Jacob Liberman

- Updated to Async 4 release
- Added final software versioning

Revision 0.3	Wednesday June 4, 2014	Jacob Liberman

- Incorporated services review edits

Revision 0.2	Thursday May 15, 2014	Jacob Liberman

- First draft text
- Updated command output sections

Revision 0.1	Monday April 14, 2014	Jacob Liberman

- Outline complete

Appendix B: Configure Storage Server

This section describes the NFS server configuration used in this reference architecture. The following script was run on all servers to configure the NFS share.

```
if [ $(hostname -s) == "nfs-server" ]
then
        yum install -y -q nfs-utils
        service nfs start
        chkconfig nfs on
        chkconfig nfslock on

        cp /etc/exports /etc/exports.orig
        groupadd -g 161 glance
        useradd -u 161 -g 161 glance
        mkdir -p /srv/rhos/glance
        chown 161.161 /srv/rhos/glance
        echo "/srv
17.31.0.0/255.255.0.0(rw,sync,no_root_squash,subtree_check,fsid=10)" >
/etc/exports

        exportfs -rav
        iptables -I INPUT -p tcp --dport 2049 -j ACCEPT
        service iptables save
        service iptables restart
        iptables -L -n | grep 2049
        showmount -e localhost
else
        mkdir -p /srv
        echo "nfs-server:/srv /srv nfs defaults 0 0" >> /etc/fstab
        iptables -I INPUT -p tcp --dport 2049 -j ACCEPT
        service iptables save
        service iptables restart
        iptables -L -n | grep 2049

        while ! (mount -av | grep srv)
        do
            sleep 10
        done
fi
```

Appendix C: Configure Shared Storage

In this reference architecture the MySQL state database runs as an active/passive HA cluster resource on the controller nodes. An iSCSI target was used for the shared storage. This appendix describes how the shared storage was discovered and configured.

1. Start and enable **iscsid** at start up.

```
[root@rhos2 ~]# service iscsid start

[root@rhos2 ~]# chkconfig iscsid on
```

2. Configure the initiator name.

```
[root@rhos2 ~]# echo "InitiatorName=iqn.1994-05.com.redhat:$(hostname
-s)" > /etc/iscsi/initiatorname.iscsi
```

3. Configure the authentication values in */etc/iscsi/iscsid.conf*.

```
node.session.auth.authmethod = CHAP
node.session.auth.username = initiatorCHAP
node.session.auth.password = examplepassword
discovery.sendtargets.auth.authmethod = CHAP
discovery.sendtargets.auth.username = initiatorCHAP
discovery.sendtargets.auth.password = examplepassword
```

4. Discover targets on the iSCSI portal.

```
[root@rhos2 ~]# iscsiadm -m discovery -t st -p 172.31.143.200
Starting iscsid:                                           [  OK  ]
172.31.143.200:3260,1 iqn.2001-05.com.equallogic:0-1cb196-e9560c201-
1830000001650c88-rhos-lun
```

5. List discovered targets.

```
[root@rhos2 ~]# iscsiadm -m node -l
Logging in to [iface: default, target: iqn.2001-05.com.equallogic:0-
1cb196-e9560c201-1830000001650c88-rhos-lun, portal: 172.31.143.200,3260]
(multiple)
Login to [iface: default, target: iqn.2001-05.com.equallogic:0-1cb196-
e9560c201-1830000001650c88-rhos-lun, portal: 172.31.143.200,3260]
successful.
```

This script was used to create a logical volume on the shared storage from a single controller node. The remaining controller nodes discover the logical volume once it is active and formatted.

```
if [ $(hostname -s) == "rhos2" ]
then
        partprobe -s
```

```
        # format the pv
        pvcreate -yv -ff /dev/sdb

        # create the VG and LV
        vgcreate mysql-vol /dev/sdb
        lvcreate -L 200G -n mysql-lv mysql-vol
        vgs
        lvs
        lvdisplay mysql-vol/mysql-lv

        # format the LV
        mkfs.ext4 /dev/mysql-vol/mysql-lv

        # disable the VG
        vgchange -aen /dev/mysql-vol

        # create a local volume list
        sed -i '/# volume_list =/a volume_list = [ "myvg", "@rhos2" ]'
/etc/lvm/lvm.conf
else
        while ! ( pvscan | grep mysql-vol)
        do
                sleep 10
        done

        if [ $(hostname -s) == "rhos3" ]
        then
                # create a local volume list
                sed -i '/# volume_list =/a volume_list = [ "myvg",
"@rhos3" ]' /etc/lvm/lvm.conf
        else
                sed -i '/# volume_list =/a volume_list = [ "myvg",
"@rhos4" ]' /etc/lvm/lvm.conf
        fi

fi
```

Script output from the controller node responsible for creating the logical volume.

```
  Warning: WARNING: the kernel failed to re-read the partition table on
  /dev/sda (Device or resource busy).  As a result, it may not reflect all
  of your changes until after reboot.
      Set up physical volume for "/dev/sdb" with 1048596480 available
  sectors
      Zeroing start of device /dev/sdb
      Writing physical volume data to disk "/dev/sdb"
    Physical volume "/dev/sdb" successfully created
    Volume group "mysql-vol" successfully created
    Logical volume "mysql-lv" created
    VG        #PV #LV #SN Attr   VSize   VFree
    mysql-vol   1   1   0 wz--n- 500.01g 300.01g
    myvg        1   1   0 wz--n- 134.94g      0
    LV       VG        Attr       LSize   Pool Origin Data%  Move Log Cpy
  %Sync Convert
    mysql-lv mysql-vol -wi-a----- 200.00g
```

```
  rootvol  myvg      -wi-ao---- 134.94g
  --- Logical volume ---
  LV Path                /dev/mysql-vol/mysql-lv
  LV Name                mysql-lv
  VG Name                mysql-vol
  LV UUID                EanA1x-5WB8-Bcmt-osGu-wKjZ-QxKA-Nkf0eb
  LV Write Access        read/write
  LV Creation host, time rhos2.cloud.lab.eng.bos.redhat.com, 2014-04-25
11:11:50 -0500
  LV Status              available
  # open                 0
  LV Size                200.00 GiB
  Current LE             51200
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:1

mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
13107200 inodes, 52428800 blocks
2621440 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
1600 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632,
2654208,
      4096000, 7962624, 11239424, 20480000, 23887872

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 22 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
  0 logical volume(s) in volume group "mysql-vol" now active
```

NOTE: The *volume_list* entry in */etc/lvm/lvm.conf* prevents the cluster nodes from activating the shared volume group at boot. This ensures that the cluster service can start the volume group only on the active controller node in the cluster.