Red Hat Reference Architecture Series

OpenShift Enterprise 2.0 on Red Hat OpenStack Platform 4

Rapidly Deploying a Highly Available OpenShift Enterprise 2.0 Environment Using Heat Templates

Aaron Weitekamp, Sr. Software Engineer

RHCSA

Scott Collier, Sr. Principal Software Engineer

RHCA

Version 1.0

March 2014

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

Comments and Feedback

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

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

Staying In Touch

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

Like us on Facebook:

https://www.facebook.com/rhrefarch

Follow us on Twitter:

https://twitter.com/RedHatRefArch

Plus us on Google+:

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

Table of Contents

1 Executive Summary

The purpose of this reference architecture is to demonstrate how a highly available OpenShift Enterprise 2.0 environment may be rapidly deployed using OpenStack Heat templates. A Heat Orchestration Template (HOT) was developed specifically for this reference architecture. The template demonstrates automating the orchestration of OpenStack resources needed for a highly available OpenShift Enterprise environment.

Virtual machine (VM) image preparation using Disk Image Builder is also demonstrated as a way to reduce the time to deploy OpenShift Enterprise and node scaling. A dynamic Domain Name Server (DNS) server is integrated into the OpenShift deployment. Node gears are stored on attached cinder volumes to provide flexible recovery and backup options.

The design decisions made are consistent with the Heat stack deployment model where a multi-instance application is deployed and configured as an integrated stack. Heat facilitates the deployment of an OpenShift Enterprise environment in a manner that is modular, yet self-contained. In a modular environment services are redundant or replicated and can easily be replaced. In a self-contained environment all the required parts are deployed with minimal external dependencies.

2 Components Overview

2.1 OpenStack

2.1.1 What is OpenStack?

OpenStack is an open source project for building a private or public Infrastructure-as-a-Service (IaaS) cloud running on standard hardware, bringing public, cloud-like capabilities to the datacenter. OpenStack is a cloud operating system that controls all the infrastructure—compute, storage, and networking—resources throughout a datacenter, all managed through a central dashboard called Horizon as shown in **Illustration 1: Red Hat Enterprise Linux OpenStack Platform Resources**.

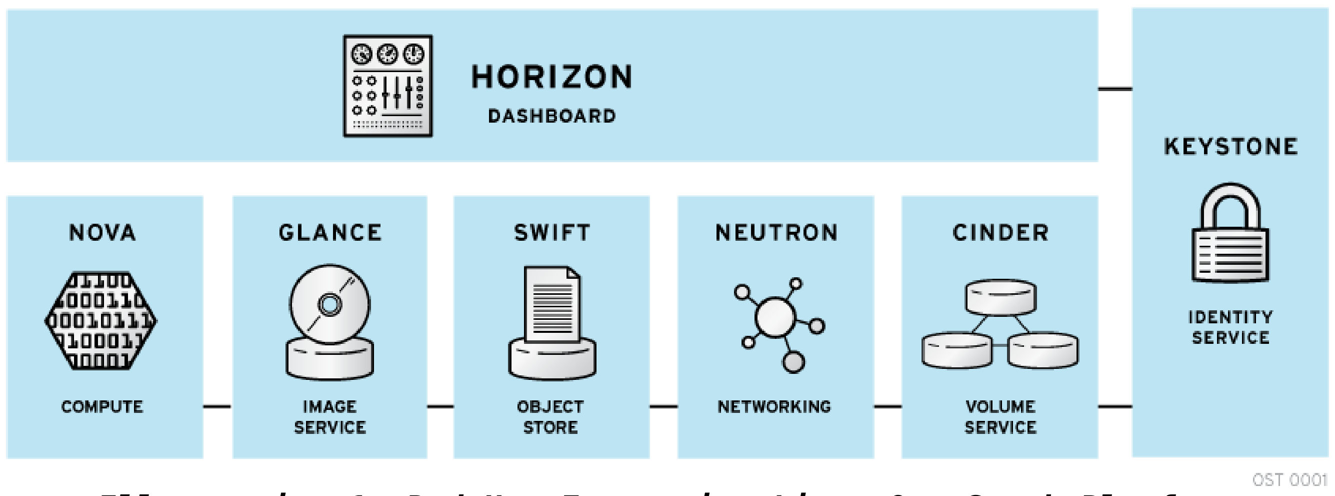


Illustration 1: Red Hat Enterprise Linux OpenStack Platform Resources

2.1.2 Red Hat Enterprise Linux OpenStack Platform 4.0 ("Havana")

Red Hat Enterprise Linux OpenStack Platform delivers an integrated foundation to create, deploy, and scale a secure and reliable public or private OpenStack cloud. It delivers a cloud platform built on Red Hat Enterprise Linux, optimized for and integrated with Red Hat's OpenStack technologies, providing the agility to scale and quickly meet customer demands without compromising on availability, security, or performance. Additionally, when combined with the Red Hat Storage Server 2 add-on, one gains an integrated and distributed highly available cloud storage platform across public and private clouds.

Red Hat Enterprise Linux OpenStack Platform 4 is built from the combination of Red Hat Enterprise Linux 6.5 and the latest Red Hat OpenStack technology, which is an enterprise-hardened version of the community "Havana" release. This version provide core features and functions from the community as well as additional innovations by Red Hat.

Specifically, Red Hat Enterprise Linux OpenStack Platform 4 now includes *Heat*, a template-based orchestration service for provisioning infrastructure resources. It also includes *Ceilometer*, which collects and stores metering and usage data and makes that data available via APIs to custom billing systems. Additionally, Red Hat Enterprise Linux OpenStack Platform 4 now utilizes the highly scalable Foreman provisioning tool for enterprise deployments, which includes bare-metal provisioning for new compute nodes. Finally, in an effort to provide a more seamless infrastructure to customers, it is now integrated with additional Red Hat infrastructure and cloud management tools, including Red Hat CloudForms for system-wide management and Red Hat Storage to provide optional object, block, and image storage services.

Key Benefits:

- **Improve service-level agreements (SLAs)** with faster IT service delivery. Red Hat Enterprise Linux OpenStack Platform delivers a Red Hat OpenStack distribution with the performance, security, and scalability of Red Hat Enterprise Linux, letting one focus on delivering the services customers want instead of the underlying operating platform.
- **Avoid vendor lock-in** by moving to open technologies while maintaining existing infrastructure investments.
- **Move from traditional workloads** to cloud-enabled workloads on an enterprise's terms and timeline, and as applications require.
- **Take advantage of broad application support.** Red Hat Enterprise Linux running as guest VMs provides a stable application development platform with many independent software vendor (ISV) certifications so cloud applications can be rapidly built and deployed.
- **Bring security to the cloud.** Rely on SELinux military-grade security technology of Red Hat Enterprise Linux to prevent intrusions and protecting data when running in public or private clouds.

2.1.3 Heat

The OpenStack Heat is an orchestration service that manages the life-cycle of applications within an OpenStack environment using templates. Heat is capable of deploying multi-instance applications called stacks and managing application lifecycle.

Heat comprises a number of Python applications.

- **`heat`**: The Heat tool is a command line interface (CLI) which communicates with the heat-api to execute OpenStack Heat Orchestration APIs.
- **`heat-api`**: The heat-api component provides an OpenStack-native ReST API that processes API requests by sending them to the heat-engine over RPC.
- **`heat-engine`**: The Heat engine does the main work of orchestrating the launch of templates and providing events back to the API consumer.
- **`heat-api-cfn`**: The heat-api-cfn component provides an AWS-style Query API that is

compatible with AWS CloudFormation and processes API requests by sending them to the heat-engine over RPC.

Heat Templates

Heat templates are written in a declarative format. A template defines what resources to deploy rather than how to deploy those resources. This is similar to the approach used by popular configuration tools such as Puppet, Ansible and Chef. Configuration tools focus on the *CONFIGURATION* of a system, whereas Heat focuses more on resource *PROVISIONING* and relies on **`cloud-init`** scripting to handle system configuration. One may use **`cloud-init`** to integrate other configuration tools into a Heat stack deployment.

Heat is compatible with AWS CloudFormation[1] (CFN). Heat templates may mix both AWS CFN and native OpenStack resources. This allows for template portability between AWS, OpenStack and other compatible cloud providers. A template may create and configure a large list of resources[2] thus supporting complex application stacks.

Heat supports two template specifications. CFN templates follow AWS CFN specification using JavaScript Object Notation (JSON) syntax. CFN templates may also be written in YAML, which is interpreted as JSON at runtime. Heat Orchestration Templates (HOT) are an optimization of the Amazon CFN format focused around ease of use and simplicity declared in YAML rather then JSON. See **Table 2.1.3.1: Identifying Heat Templates**.

Template	Syntax	First Line Identifier (YAML only)
CFN	YAML or JSON	HeatTemplateFormatVersion: '2012-12-12'
HOT	YAML	heat_template_version: 2013-05-23

Table 2.1.3.1: Identifying Heat Templates

HOT-based templates provide a simpler, easier to read syntax over CFN organized in five sections:

1. **description**: human-readable description
2. **parameter groups**: a grouped list of parameters to organize large parameter sets
3. **parameters**: defines parameters, providing description, data type (string, number, etc), default value and other options.
4. **resources**: defines all the resources managed by the template. Resources are the components that are deployed: a server, floating IP, etc.
5. **outputs**: Variables or other information about a stack deployment. In a nested stack output values may be accessed from the parent template.

Heat templates vary widely in complexity. Typically templates are written by an application subject matter expert since application requirements determine the complexity of the template.

Note: CFN Heat templates are fully supported in Red Hat Enterprise Linux OpenStack Platform 4. HOT-based templates are released as tech preview. Full support for HOT is

1 http://aws.amazon.com/cloudformation/
2 http://docs.openstack.org/developer/heat/template_guide/

expected in the *Icehouse* release of OpenStack, at which point HOT-based templates will be the preferred format going forward.

2.1.4 Disk Image Builder

Disk Image Builder[3] (DIB) is a VM image building tool provided with Red Hat Enterprise Linux OpenStack Platform. It provides a method to prepare guest images for application deployment. The DIB model is to build virtual machine images that have all the necessary packages for specific applications, allowing application deployments to be disconnected from the package source network. Instead of maintaining a few generic base images, the DIB workflow encourages maintaining many images, one for each type of application. With this model automating image building becomes important. DIB is designed to be part of an automated continuous integration (CI) image building process. The goal of this model is to maintain a library of “golden” images that are continuously maintained for deployment at all times. The core of the CI model is software updates trigger a new VM image build process. To reduce image building time DIB resources are cached to be available for future image builds.

DIB saves a new, modified qcow2 image after processing. Image builds are parameterized by mixing in elements from multiple sources. Custom elements may be created and defined with dependencies.

Preparing images with disk image builder is not required for this reference architecture but it greatly reduces the time to deploy the OpenShift application stack. The base guest image from Red Hat may be used unaltered if deployment time is not a concern or if an organization wishes to keep image maintenance simple.

2.1.5 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`**.

OpenStack Networking is composed of several services.

- **`neutron-server`** exposes the API and responds to user requests.
- **`neutron-l3-agent`** provides L3 functionality, such as routing, through interaction with the other networking plugins and agents.
- **`neutron-dhcp-agent`** provides DHCP to tenant networks.
- There are also a series of network agents that perform local networking configuration for the node’s VMs. This reference architecture is based on the **`openvswitch`** plugin, which uses the **`neutron-openvswitch-agent.`**

3 https://github.com/openstack/diskimage-builder

2.1.6 Load Balancer as a Service ("LBaaS")

OpenStack Load Balancer as a Service (*LBaaS*) is a *Neutron* extension that manages traffic for services running on instances. Instead of having a dedicated virtual machine for load balancing, LBaaS uses an agent model. The **`neutron-lbaas-agent`** daemon manages the load balancer provider. In this environment **`HA Proxy`** is used but other load balancer technologies may be used as support is enabled. LBaaS supports common protocol types *ICMP*, *TCP*, *HTTP* and *HTTPS*.

LBaaS has several components.

- **`Health Monitor`** polls the real servers and supports polling parameters type (e.g. *HTTPS*), expected response code (e.g. *200*), delay, timeout and retry.
- **`Pool`** defines the list of servers to be load balanced and parameters such as method algorithm (e.g. *ROUND_ROBIN*) and service provider (e.g. **`HA Proxy`**).
- **`Virtual IP`** (*VIP*) is the interface that a floating IP address is associated with.

See **Figure 2.1.6.1: Load Balancer as a Service (LBaaS)** for a graphical depiction of LBaaS.

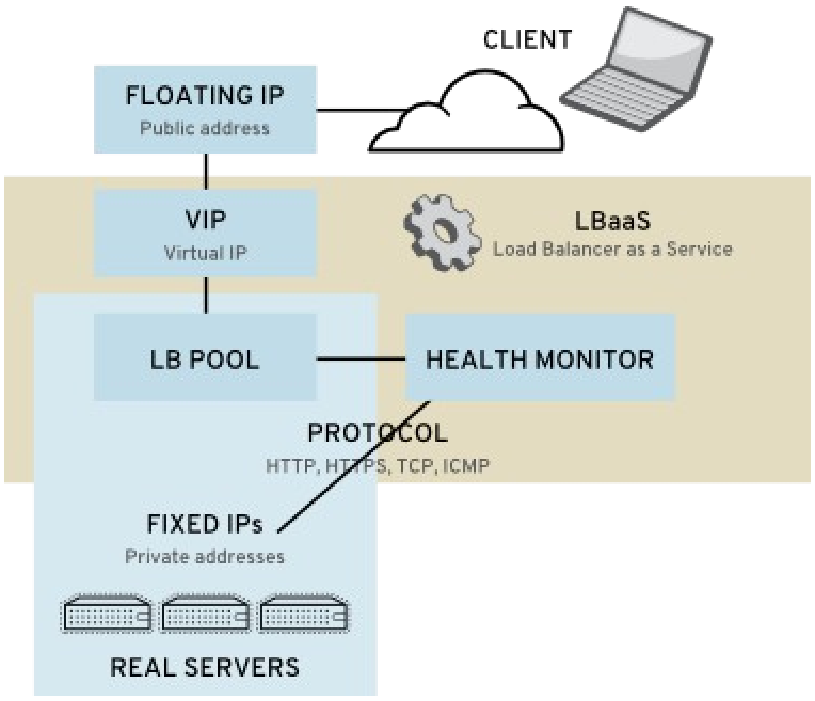


Figure 2.1.6.1: Load Balancer as a Service (LBaaS)

2.1.7 Block Storage ("Cinder")

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

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

The Cinder service is comprised of several components.

- **`openstack-cinder-api`** responds to service requests.
- **`openstack-cinder-scheduler`** assigns tasks to the queue.
- **`openstack-cinder-volume`** service interacts with various storage providers to allocate block storage for VMs, including Red Hat Storage Server[4].

2.2 OpenShift

OpenShift is a Platform as a Service (PaaS) solution from Red Hat. OpenShift provides a multi-language, auto-scaling, self-service, elastic cloud application platform built on a proven stack of Red Hat technologies. The OpenShift PaaS services are available as both a hosted service and on-premise PaaS product offering. OpenShift Enterprise is Red Hat's on-premise PaaS software solution that enables customers to deploy their own Private or Hybrid PaaS environment.

OpenShift Enterprise enables administrators to more quickly serve the needs of application developers by deploying a PaaS platform that streamlines the application service delivery process. OpenShift Enterprise enables administrators and developers to more quickly serve their needs by standardizing and accelerating developer work-flows. OpenShift Enterprise does this in a highly-efficient, fine-grained, multi-tenant cloud architecture that maximizes infrastructure utilization. This provides application developers with self-service access so they can easily deploy applications on-demand leveraging the languages, frameworks, and tools of their choosing. It ultimately increases developer efficiency and agility by allowing them to focus on what they do best, writing code, and in turn enables them to better meet the demands of the business.

OpenShift Enterprise is built on the strength of a proven stack of Red Hat technologies. It provides the freedom for developers and administrators to work the way they want to work. For developers, OpenShift Enterprise supports Java, Ruby, PHP, Python, and Perl programming languages with associated frameworks as well as additional platform services like MySQL and PostgreSQL databases, Jenkins Continuous Integration server, and more. OpenShift Enterprise "cartridges" provide these platform services. Cartridges are extensible, enabling customers to extend OpenShift to add their own custom services. Developers can access OpenShift Enterprise via a rich CLI, web console, RESTful API or Eclipse integrated development environment (IDE). For IT administrators, OpenShift Enterprise runs on the trusted Red Hat Enterprise Linux platform and leverages technologies such as SELinux and

4 http://www.redhat.com/products/storage-server/

Cgroups to provide a secure and scalable multi-tenant environment, that allows efficient infrastructure utilization supporting application development and deployment. OpenShift also provides standards-based application run-time systems like JBoss Enterprise Application Platform, Tomcat and Apache. OpenShift Enterprise can be deployed on-premise in a customer's data center or private cloud on their choice of virtualization platform or cloud provider. In addition, developers can work from a variety of workstations including Red Hat Linux, Mac and Windows. OpenShift Enterprise is open source, based on the upstream OpenShift Origin project, that helps drive innovation and eliminate lock-in.

2.2.1 Broker

The broker is the orchestration mechanism for all application management activities. It is responsible for the following tasks:

- Managing user logins
- Dynamically updating DNS
- Application state
- Application orchestration
- Services and operations

In addition to being able to contact the broker directly via a RESTful API, the following methods may also be used

- Web console
- CLI tools
- Eclipse JBoss Developer Studio

OpenShift Enterprise is composed of several components that make the above tasks possible. This reference architecture focuses on distributing OpenShift Enterprise for the purposes of high availability. In order to do that, the following services are configured for redundancy:

- MongoDB
- ActiveMQ

MongoDB maintains application state for the environment. Where high availability is required MongoDB is set up as a replica set spanning multiple broker hosts so application state is not lost if a broker hosts is down.

ActiveMQ is the message broker providing communication between broker and node hosts. Where high availability is required ActiveMQ is set up as a messaging pool across multiple broker hosts for redundancy so messages are not lost if a broker host fails. Messages are sent using Mcollective. Each broker host runs an Mcollective client. Each node host runs an Mcollective server which subscribes to messaging topics and listens for incoming messages. Product documentation for OpenShift Enterprise can be found on Red Hat's customer portal[5].

5 https://access.redhat.com/site/documentation/en-US/OpenShift_Enterprise/

2.2.2 Node

An OpenShift node is a host that runs the applications. There can be many nodes that are deployed that a broker manages. A node supports gears that contain applications. One of the features of OpenShift is that it is a true multi-tenancy environment. This is accomplished by using SELinux and Cgroup restrictions so as to separate each applications' resources and data.

It is not recommended to combine broker and node roles on the same host. This reference architecture deploys the node services onto a set of dedicated node hosts.

2.3 BIND

Berkeley Internet Name Daemon (BIND) is an implementation of the DNS protocols. BIND enables a human or computer to look-up another computer on the basis of a name. BIND contains three parts:

- DNS Server – answers queries that are sent to it
- DNS Resolver Library – library that can be added to other programs that provides the ability to resolve names ensuring DNS standards are being followed
- Tools for Testing Servers – Tools used for testing, such as **`dig`**, **`nslookup`** and **`host`**.

BIND is an integral part of a successful OpenShift Enterprise deployment / environment. OpenShift Enterprise uses a RubyGem called **`dnsruby`** to issue dynamic updates to the DNS environment. The requirement that allows this functionality is that the DNS environment supports *RFC 2136* (Dynamic DNS Updates). Any DNS implementation that supports *RFC 2136* will work in a OpenShift Environment, however this reference architecture uses BIND.

2.3.1 Dynamic DNS

There are many ways to configure DNS. In this reference architecture dynamic DNS is approached as an integrated part of the OpenShift environment so a DNS server is configured on *broker1*.

During the initial OpenShift DNS configuration, an *A* record is entered for each OpenShift server to associate hostname and IP address. When a new application cartridge is created the DNS server is dynamically updated with a *CNAME* record that associates the application cartridge name and the node IP address serving the application.

```
$ORIGIN ose.example.com.
broker                  A       10.16.138.131
broker1                 A       10.16.138.130
broker2                 A       10.16.138.132
broker3                 A       10.16.138.129
node1                   A       10.16.138.134
node2                   A       10.16.138.135
node3                   A       10.16.138.133
app1                    CNAME   node3
app2                    CNAME   node2
```

In this reference architecture the dynamic DNS server on broker1 is the authoritative master for the openshift zone. An *upstream* non-authoritative DNS slave is configured for the OpenShift zone. The zone master is configured to notify the upstream DNS slave when the zone file has changed. A zone transfer occurs when the DNS zone master is updated dynamically, for example when a new application is deployed.

1. CNAME record is recorded
2. Zone file serial number is incremented
3. Upstream zone slave is notified
4. Upstream zone slave receives the updated zone file

The DNS zone slave can now resolve queries to the new application.

In this configuration dynamic DNS is a single point of failure. If broker1 fails then dynamic updates will fail but existing applications continue to resolve from the non-authoritative zone slave. If highly available dynamic DNS is required then a DNS server supporting multi-master replication is recommended. See documentation for Red Hat Identity Management[6].

In this reference architecture DNS resolves for forward lookups only—i.e. lookups by hostname. Reverse lookups (i.e. lookups by IP address) are not configured.

6 https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Identity_Management_Guide/index.html

3 Reference Architecture Configuration

3.1 Environment

This reference architecture environment consists of all the components required to build a OpenShift Enterprise PaaS in a distributed fashion. The information is shared throughout this document.

The environment used in this reference architecture is depicted in **Figure 3.1.1: Reference Architecture Environment**. This environment consists of three physical servers and eight VMs, although this is configurable. At a minimum the Heat template-based deployment demonstrated below requires six VMs.

Red Hat Enterprise OpenStack Platform is deployed on the three physical machines. The *controller* server hosts all the OpenStack supporting services, including *Horizon*, *Neutron*, *Glance* and *Keystone* services. *LBaaS* is also served from the controller machine as part of Neutron. Two physical machines host the *Nova* compute service, allowing for the OpenShift VM environment to be distributed between the compute nodes using availability zones.

The OpenShift environment distributes MongoDB and ActiveMQ as highly available (HA) services on three broker VMs. The database (state) and messaging (communication) infrastructure will not be compromised if a broker VM fails. Note, with two MongoDB nodes running, database reads succeed but not writes. See **Figure 3.1.1: Reference Architecture Environment** for a more detailed view of the deployed OpenShift environment.

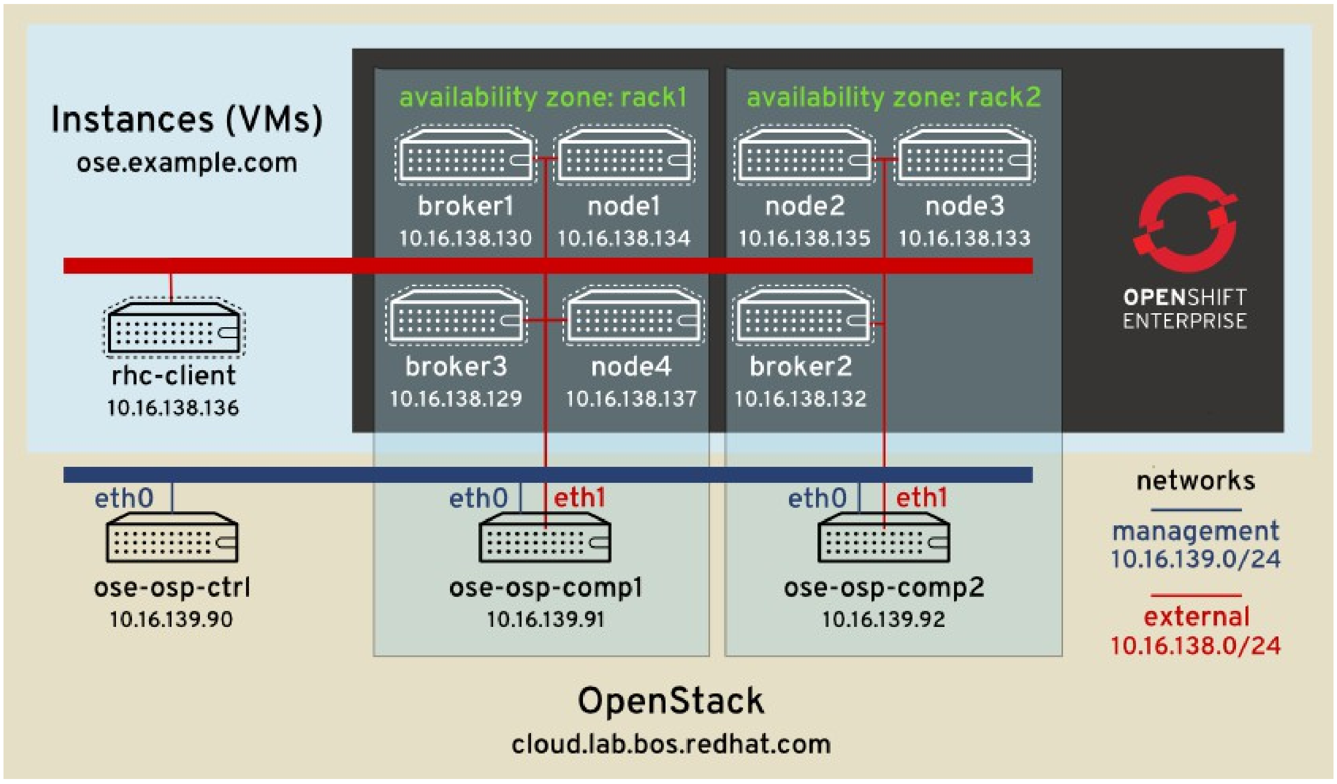


Figure 3.1.1: Reference Architecture Environment

3.2 Hardware

The three OpenStack servers are configured with the same hardware platform type.

Hardware	Component	Specifications
IBM System x3550 M3	Processor	Intel(R) Xeon(R) CPU E5645 @ 2.40GHz, 8 cores
	NICs	2, Broadcom Corporation NetXtreme II BCM5709 Gigabit Ethernet
	Memory	35GB
	Disk	600GB

Table 3.2.1: Server Hardware

The configuration of the physical servers used in this reference environment are listed in the following table.

Hostname	Role	Services	IP Address
ose-osp-ctrl	OpenStack controller	horizon glance cinder neutron	10.16.139.90
ose-osp-comp1	OpenStack compute	nova	10.16.139.91
ose-osp-comp2	OpenStack compute	nova	10.16.139.92

Table 3.2.2: Physical Server Configuration

3.3 Virtual Machines

3.3.1 Sizing

Sizing VMs for OpenShift nodes is highly dependent on the number and type of applications running on the node. Memory and CPU allocations should be adjusted as needed to support application and usage needs.

OpenShift broker VMs are under much less variable load. The three-broker HA environment can support a large OpenShift deployment.

Hostname	Disk	Memory	CPU
rhc-client	20GB	2GB	1 Virtual Core
broker{1,2,3}	15GB	4GB	2 Virtual Core
node{1,2,3,4,5}	15GB	4GB	2 Virtual Core

Table 3.3.1.1: Virtual Machine Sizing

3.3.2 Addressing

Note: VM addresses are allocated dynamically from a pool of available addresses during the Heat stack deployment.

Hostname	Role	IP Address
rhc-client	client	10.16.138.136
broker1	OpenShift broker	10.16.138.130
broker2		10.16.138.132
broker3		10.16.138.129
node1	OpenShift node	10.16.138.134
node2		10.16.138.135
node3		10.16.138.133
node4		10.16.138.137

Table 3.3.2.1: Virtual Machine IP Addresses

3.4 Software

3.4.1 Software Versions

Refer to **Appendix E: Software Versions** for a list of software packages used in this reference architecture.

3.4.2 Configuration Files

The configuration files and scripts used to reproduce this reference architecture are included on the Red Hat customer portal[7]. A log-in is required to access the configuration files. **Appendix F: Configuration Files** contains a list of files included.

7 https://access.redhat.com/site/node/771923/40/1

3.4.3 Red Hat Network

This table lists the RHN channels used for Red Hat Products. Content is served from Red Hat Satellite in the lab.

Product	Channels
Red Hat Enterprise Linux OpenStack Platform 4	rhel-x86_64-server-6 rhel-x86_64-server-6-mrg-messaging-2 rhel-x86_64-server-6-ost-4 rhel-x86_64-server-6-rhscl-1 rhel-x86_64-server-lb-6 rhel-x86_64-server-optional-6 rhel-x86_64-server-rh-common-6
OpenShift Enterprise 2.0	jb-ews-2-x86_64-server-6-rpm jbappplatform-6-x86_64-server-6-rpm rhel-x86_64-rhev-agent-6-server rhel-x86_64-server-6 rhel-x86_64-server-6-ose-2.0-infrastructure rhel-x86_64-server-6-ose-2.0-jbosseap rhel-x86_64-server-6-ose-2.0-node rhel-x86_64-server-6-ose-2.0-rhc rhel-x86_64-server-6-rhscl-1 rhn-tools-rhel-x86_64-server-6

Table 3.4.3.1: RHN Channels

4 Deploying the Environment

This section describes the procedures used preparing the environment. This does not include installation of Red Hat Enterprise Linux OpenStack Platform from the beginning. Only specific configurations necessary for this reference environment are discussed.

> **Note:** Customer Portal users may download the configuration files used in this environment. See **Appendix F: Configuration Files** for a list of included files.

4.1 Red Hat Enterprise Linux OpenStack Platform Configuration

The OpenStack configuration discussed for the reference environment include configuring:

- Neutron networking
- Load Balancer as a Service (LBaaS)
- Cinder storage
- Flavors and Zones
- Image building

4.1.1 Neutron Networking

The reference environment utilizes two physical network interfaces per server. The management network interface provides access to manage OpenStack resources. The external network interface carries OpenStack API traffic as well as VM traffic.

The *private* and *public* virtual networks are defined using Neutron. The public network is configured with a pool of floating IP addresses for external access to instances. Perform the following steps on the OpenStack *controller* running neutron.

1. Source the keystone file and list the empty neutron network and subnet.

```
[ose-osp-ctrl ~]# source keystonerc_admin
[ose-osp-ctrl ~(keystone_admin)]# neutron net-list

# neutron subnet-list
```

2. Create a private neutron network.

```
# neutron net-create private
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 68579291-519f-4c3f-88a1-1b6a01be4e23 |
| name                      | private                              |
| provider:network_type     | vlan                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  | 1111                                   |
| shared                    | False                                  |
| status                    | ACTIVE                                 |
| subnets                   |                                        |
| tenant_id                 | 8ac0aa9d7a284221aac7a9729e0846ce       |
+---------------------------+----------------------------------------+
```

3. Create a private neutron subnet.

```
# neutron subnet-create \
--name root-subnet private 10.0.6.0/24 \
--dns-nameserver 10.16.143.247
Created a new subnet:
+------------------+--------------------------------------------+
| Field            | Value                                      |
+------------------+--------------------------------------------+
| allocation_pools | {"start": "10.0.6.2", "end": "10.0.6.254"} |
| cidr             | 10.0.6.0/24                                |
| dns_nameservers  | 10.16.143.247                              |
| enable_dhcp      | True                                       |
| gateway_ip       | 10.0.6.1                                   |
| host_routes      |                                            |
| id               | 936232a2-05f1-4bfd-b093-9cebc44056c9       |
| ip_version       | 4                                          |
| name             | root-subnet                                |
| network_id       | 68579291-519f-4c3f-88a1-1b6a01be4e23       |
| tenant_id        | 8ac0aa9d7a284221aac7a9729e0846ce           |
+------------------+--------------------------------------------+
```

4. Create a neutron router.

```
# neutron router-create router1
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 9aa3c260-16ee-49e8-a3f3-fc98a264e213 |
| name                  | router1                              |
| status                | ACTIVE                               |
| tenant_id             | 8ac0aa9d7a284221aac7a9729e0846ce     |
+-----------------------+--------------------------------------+
```

5. List the neutron subnet, noting the ID, and use the subnet ID to add a router interface.

```
# neutron subnet-list
+--------------------------------------+-------------+-------------
+--------------------------------------------+
| id                                   | name        | cidr        |
allocation_pools                             |
+--------------------------------------+-------------+-------------
+--------------------------------------------+
| 936232a2-05f1-4bfd-b093-9cebc44056c9 | root-subnet | 10.0.6.0/24 | {"start":
"10.0.6.2", "end": "10.0.6.254"} |
+--------------------------------------+-------------+-------------
+--------------------------------------------+
# neutron router-interface-add router1 936232a2-05f1-4bfd-b093-
9cebc44056c9
Added interface b2043eb9-54ee-4171-bcf6-5083f0be9a2f to router router1.
```

6. Create a public neutron network and subnet.

```
# neutron net-create public \
--router:external=True \
--provider:network_type flat \
--provider:physical_network physnet1
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 1e103795-4c72-4905-987e-0f434fce77f9 |
| name                      | public                               |
| provider:network_type     | flat                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 8ac0aa9d7a284221aac7a9729e0846ce     |
+---------------------------+--------------------------------------+
# neutron subnet-create \
--name public \
--gateway 10.16.143.254 \
--allocation-pool start=10.16.138.128,end=10.16.138.152 \
public 10.16.136.0/21 \
--enable_dhcp=False
Created a new subnet:
+------------------+----------------------------------------------------+
| Field            | Value                                              |
+------------------+----------------------------------------------------+
| allocation_pools | {"start": "10.16.138.128", "end": "10.16.138.152"} |
| cidr             | 10.16.136.0/21                                     |
| dns_nameservers  |                                                    |
| enable_dhcp      | False                                              |
| gateway_ip       | 10.16.143.254                                      |
| host_routes      |                                                    |
| id               | 24761a80-166f-4193-9a45-bdc0cf5cc553               |
| ip_version       | 4                                                  |
| name             | public                                             |
| network_id       | 1e103795-4c72-4905-987e-0f434fce77f9               |
| tenant_id        | 8ac0aa9d7a284221aac7a9729e0846ce                   |
+------------------+----------------------------------------------------+
```

7. List the neutron networks, noting the public network ID, and set the router gateway to use the public network.

```
# neutron net-list
+--------------------------------------+---------
+-----------------------------------------------------+
| id                                   | name    | subnets
|
+--------------------------------------+---------
+-----------------------------------------------------+
| 1e103795-4c72-4905-987e-0f434fce77f9 | public  | 24761a80-166f-4193-9a45-
bdc0cf5cc553 10.16.136.0/21 |
| 68579291-519f-4c3f-88a1-1b6a01be4e23 | private | 936232a2-05f1-4bfd-b093-
```

```
9cebc44056c9 10.0.6.0/24    |
+--------------------------------------+---------
+--------------------------------------------------+
# neutron router-gateway-set router1 1e103795-4c72-4905-987e-
0f434fce77f9
Set gateway for router router1
# neutron router-show router1
+-----------------------
+-----------------------------------------------------------------------------+
| Field                 | Value
|
+-----------------------
+-----------------------------------------------------------------------------+
| admin_state_up        | True
|
| external_gateway_info | {"network_id": "1e103795-4c72-4905-987e-0f434fce77f9",
"enable_snat": true} |
| id                    | 9aa3c260-16ee-49e8-a3f3-fc98a264e213
|
| name                  | router1
|
| routes                |
|
| status                | ACTIVE
|
| tenant_id             | 8ac0aa9d7a284221aac7a9729e0846ce
|
+-----------------------
+-----------------------------------------------------------------------------+
```

Refer to *Red Hat Enterprise Linux OpenStack Platform 4: End User Guide*[8] and *Red Hat Enterprise Linux OpenStack Platform 4: Installation and Configuration Guide*[9] for additional information regarding installing and configuring network services for Red Hat Enterprise Linux OpenStack Platform.

4.1.2 LBaaS

In this reference environment LBaaS is configured to use HA Proxy. HA Proxy requires an entitlement to the *Red Hat Load Balancer Add-On*.

Perform the following steps on the OpenStack *controller* running neutron.

1. Add the *Load Balancer Add-On* channel

```
# rhn-channel --user admin --password BADPASS -a -c rhel-x86_64-server-
lb-6
```

2. Install **haproxy**

```
# yum -y install haproxy
```

3. Configure LBaaS. In this step the interface, device driver and user_group for the *lbaas_agent* is configured, and the Neutron service plugin is configured.

8 https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/4/html/End_User_Guide/dashboard_create_networks.html

9 https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/4/html/Installation_and_Configuration_Guide/chap-Installing_the_OpenStack_Networking_Service.html

```
# echo "interface_driver =
neutron.agent.linux.interface.OVSInterfaceDriver" >>
/etc/neutron/lbaas_agent.ini
# echo "device_driver =
neutron.services.loadbalancer.drivers.haproxy.namespace_driver.HaproxyNS
Driver" >> /etc/neutron/lbaas_agent.ini
# echo "user_group = haproxy" >> /etc/neutron/lbaas_agent.ini
sed -i 's/# service_plugins =/service_plugins =
neutron.services.loadbalancer.plugin.LoadBalancerPlugin/'
/etc/neutron/neutron.conf
# sed -i "s/'enable_lb': False/'enable_lb': True/" /etc/openstack-
dashboard/local_settings
# service httpd restart
Stopping httpd:                                            [  OK  ]
Starting httpd:                                            [  OK  ]
# service neutron-lbaas-agent start
Starting neutron-lbaas-agent:                              [  OK  ]
# service neutron-server restart
Stopping neutron:                                          [  OK  ]
Starting neutron:                                          [  OK  ]
# chkconfig httpd on
# chkconfig neutron-lbaas-agent on
```

Note: Do not start **haproxy**. LBaaS manages **haproxy** so it is important that the **haproxy** service is not started manually or configured to start on boot. Below is expected output before an LBaaS pool is created.

```
# chkconfig --list haproxy
haproxy         0:off 1:off 2:off 3:off 4:off 5:off 6:off
# service haproxy status
haproxy is stopped
```

Later, when LBaaS is running and properly managing **haproxy**, the service status will return the *pid* and *running* status.

```
# service haproxy status
haproxy (pid 5969) is running...
```

4.1.3 Cinder Block Storage

There are many backend storage options for block storage on Red Hat Enterprise Linux OpenStack Platform. For the reference environment the *controller* host is configured as a Cinder storage device using the default implementation. By default Cinder uses logical volume management (LVM), creating a volume group named "cinder-volumes" on the specified hosts. Storage is allocated (320 GB) by including *CONFIG_CINDER_VOLUMES_SIZE=320G* in the **packstack** answerfile. The */etc/cinder/cinder.conf* file is provided in the files associated with this reference architecture. Refer to the Red Hat Enterprise Linux OpenStack Platform documentation[10] for additional Cinder configuration options.

10 https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/4/html-single/Configuration_Reference_Guide/index.html#ch_configuring-openstack-block-storage

4.1.4 Additional Configuration

Availability Zones

Availability Zones provide the ability to control the placement of instances is important for ensuring the instances are distributed in a logical manner. Without specifying availability zones instances will be placed on compute nodes based on a load algorithm. Depending on existing load and the order of instance deployment the VMs may not be placed as desired. For example, if all three broker VMs are placed on a single OpenStack compute node the environment will fail when the compute node goes down. For this reason instance placement is an important component to providing highly available services.

By default all compute nodes are assigned to the *nova* availability zone. Availability zones may be listed using the nova **availability-zone-list** command from controller node ose-osp-ctrl.cloud.lab.eng.bos.redhat.com.

```
# nova availability-zone-list
+-----------------------------------------------+-----------------------------------------
+
| Name                                          | Status
|
+-----------------------------------------------+-----------------------------------------
+
| internal                                      | available
|
| |- ose-osp-ctrl.cloud.lab.eng.bos.redhat.com |                                        |
| | |- nova-conductor                           | enabled :-) 2014-02-19T16:10:13.000000
|
| | |- nova-consoleauth                         | enabled :-) 2014-02-19T16:10:16.000000
|
| | |- nova-scheduler                           | enabled :-) 2014-02-19T16:10:16.000000
|
| | |- nova-cert                                | enabled :-) 2014-02-19T16:10:16.000000
|
| nova                                          | available
|
| |- ose-osp-comp1.cloud.lab.eng.bos.redhat.com |
|
| | |- nova-compute                             | enabled :-) 2014-02-19T16:10:13.000000
|
| |- ose-osp-comp2.cloud.lab.eng.bos.redhat.com |
|
| | |- nova-compute                             | enabled :-) 2014-02-19T16:10:13.000000
|
```

Availability zones are actually a subset of host aggregates. Host aggregates are a grouping of hosts that can be optionally assigned metadata *key: value* pairs or an availability zone. In this reference architecture aggregate metadata is not used but each host aggregate will be assigned to an availability zone.

To create host aggregates with availability zones, execute the following commands on the controller node.

```
# nova aggregate-create aggregate1 rack1
+----+------------+-------------------+-------+---------------------------------+
| Id | Name       | Availability Zone | Hosts | Metadata                        |
+----+------------+-------------------+-------+---------------------------------+
| 1  | aggregate1 | rack1             | []    | {u'availability_zone': u'rack1'} |
+----+------------+-------------------+-------+---------------------------------+
# nova aggregate-create aggregate2 rack2
+----+------------+-------------------+-------+---------------------------------+
| Id | Name       | Availability Zone | Hosts | Metadata                        |
+----+------------+-------------------+-------+---------------------------------+
| 2  | aggregate2 | rack2             | []    | {u'availability_zone': u'rack2'} |
+----+------------+-------------------+-------+---------------------------------+
# nova aggregate-add-host aggregate1 ose-osp-comp1.cloud.lab.eng.bos.redhat.com
Aggregate 1 has been successfully updated.
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
| Id | Name       | Availability Zone | Hosts                                      |
Metadata                         |
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
| 1  | aggregate1 | rack1             | [u'ose-osp-comp1.cloud.lab.eng.bos.redhat.com'] |
{u'availability_zone': u'rack1'} |
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
# nova aggregate-add-host aggregate2 ose-osp-comp2.cloud.lab.eng.bos.redhat.com
Aggregate 2 has been successfully updated.
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
| Id | Name       | Availability Zone | Hosts                                      |
Metadata                         |
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
| 2  | aggregate2 | rack2             | [u'ose-osp-comp2.cloud.lab.eng.bos.redhat.com'] |
{u'availability_zone': u'rack2'} |
+----+------------+-------------------+--------------------------------------------
+----------------------------------+
# nova availability-zone-list
+----------------------------------------------+----------------------------------------+
| Name                                         | Status                                 |
+----------------------------------------------+----------------------------------------+
| internal                                     | available                              |
| |- ose-osp-ctrl.cloud.lab.eng.bos.redhat.com |                                        |
| | |- nova-conductor                          | enabled :-) 2014-03-19T15:03:15.000000 |
| | |- nova-consoleauth                        | enabled :-) 2014-03-19T15:03:15.000000 |
| | |- nova-scheduler                          | enabled :-) 2014-03-19T15:03:15.000000 |
| | |- nova-cert                               | enabled :-) 2014-03-19T15:03:15.000000 |
| rack1                                        | available                              |
| |- ose-osp-comp1.cloud.lab.eng.bos.redhat.com |                                       |
| | |- nova-compute                            | enabled :-) 2014-03-19T15:03:15.000000 |
| rack2                                        | available                              |
| |- ose-osp-comp2.cloud.lab.eng.bos.redhat.com |                                       |
| | |- nova-compute                            | enabled :-) 2014-03-19T15:03:15.000000 |
+----------------------------------------------+----------------------------------------+
```

The availability zones *rack1* and *rack2* may now be included as parameters when deploying the Heat stack.

Custom Flavor

Flavors are OpenStack's method of describing VM specifications. Flavors define CPU, RAM and disk size. Custom flavor *m3.medium* is created to provide a mid-sized VM that has a smaller disk size to fit the VM requirements of this environment.

Perform the following on the OpenStack controller host.

1. List the default flavors.

   ```
   # nova flavor-list
   ```

```
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor |
Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True
|
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True
|
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True
|
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True
|
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True
|
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+
```

2. Since the default flavors do not meet the requirements of this reference architecture flavor *m3.medium* is created. To create a custom flavor specify an arbitrary name and ID, the memory (MB), disk (GB) and number of VCPUs.

```
# nova flavor-create --is-public true m3.medium 10 4096 15 2
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 10 | m3.medium | 4096      | 15   | 0         |      | 2     | 1.0         | True      |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
# nova flavor-show m3.medium
+----------------------------+-----------+
| Property                   | Value     |
+----------------------------+-----------+
| name                       | m3.medium |
| ram                        | 4096      |
| OS-FLV-DISABLED:disabled   | False     |
| vcpus                      | 2         |
| extra_specs                | {}        |
| swap                       |           |
| os-flavor-access:is_public | True      |
| rxtx_factor                | 1.0       |
| OS-FLV-EXT-DATA:ephemeral  | 0         |
| disk                       | 15        |
| id                         | 10        |
+----------------------------+-----------+
```

Flavor *m3.medium* may now be included as a parameter when deploying the Heat stack and other instances.

Nova Keypair

Access to instances is enabled by a key as opposed to username and password. Create the keypair and assign the proper file permissions.

```
# nova keypair-add rootkp > /root/rootkp.pem
# chmod 600 /root/rootkp.pem
```

Security Groups

OpenStack security groups limit access to specific ports and network protocols. The Heat stack will create specific security groups for both broker and node instances. The OpenShift client instance will use the default security group. In this step the default security group is configured to accept ICMP, SSH and web traffic.

```
# neutron security-group-list
+--------------------------------------+---------+-------------+
| id                                   | name    | description |
+--------------------------------------+---------+-------------+
| b8a78d36-f9bf-4685-8be8-5b2501b34185 | default | default     |
+--------------------------------------+---------+-------------+
# neutron security-group-rule-create --direction ingress --protocol icmp
default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 226af957-0900-4b5c-bc20-92e2b51af5ed |
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          | icmp                                 |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b8a78d36-f9bf-4685-8be8-5b2501b34185 |
| tenant_id         | 8ac0aa9d7a284221aac7a9729e0846ce     |
+-------------------+--------------------------------------+
# neutron security-group-rule-create --direction ingress --protocol tcp
--port_range_min 22 --port_range_max 22 default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 7ca4f08f-e7ce-4684-b849-61c095e9c272 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b8a78d36-f9bf-4685-8be8-5b2501b34185 |
| tenant_id         | 8ac0aa9d7a284221aac7a9729e0846ce     |
+-------------------+--------------------------------------+
# neutron security-group-rule-create --direction ingress --protocol tcp
--port_range_min 80 --port_range_max 80 default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 2570b0d6-a110-4acb-906d-8c14aacf752c |
| port_range_max    | 80                                   |
| port_range_min    | 80                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b8a78d36-f9bf-4685-8be8-5b2501b34185 |
| tenant_id         | 8ac0aa9d7a284221aac7a9729e0846ce     |
+-------------------+--------------------------------------+
# neutron security-group-rule-create --direction ingress --protocol tcp
--port_range_min 443 --port_range_max 443 default
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | f9fdea79-a66e-4dee-986d-00c7e4faaf5c |
| port_range_max    | 443                                  |
| port_range_min    | 443                                  |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | b8a78d36-f9bf-4685-8be8-5b2501b34185 |
| tenant_id         | 8ac0aa9d7a284221aac7a9729e0846ce     |
+-------------------+--------------------------------------+
```

The Heat templates create additional security groups during the deployment process. Unique security groups rules are defined for the OpenShift Enterprise broker and node hosts.

> **Note:** Due to a bug[11] in the OpenShift Enterprise highly available Heat template, egress rules are not configured properly. See section **4.2.1.3 Cloud-init user_data** for the manual workaround.

The following tables list all security group rules used in this environment.

Default

Direction	Protocol	Port
Ingress	ICMP	N/A
Ingress	TCP	22
Ingress	TCP	80
Ingress	TCP	443
Egress	All	All

Table 4.1.4.1: Default security group rules

11 https://bugzilla.redhat.com/show_bug.cgi?id=1080638

Broker hosts—For reference only; rules created by Heat template

Direction	Protocol	Port
Ingress	UDP	53
Ingress	TCP	53
Ingress	TCP	80
Ingress	TCP	443
Ingress	TCP	27017
Ingress	TCP	61613
Ingress	TCP	61616
Egress	All	All

Table 4.1.4.2: Broker security group rules

Node hosts—For reference only; rules created by Heat template

Direction	Protocol	Port
Ingress	UDP	53
Ingress	TCP	53
Ingress	TCP	80
Ingress	TCP	443
Ingress	TCP	8000
Ingress	TCP	8443
Ingress	TCP	2303-2308
Ingress	TCP	35531-65535
Ingress	TCP	27017
Egress	All	All

Table 4.1.4.3: Node security group rules

4.1.5 Image Preparation with Disk Image Builder

In this reference architecture DIB is used to prepare the virtual machine images for OpenStack deployment. The base image used is the Red Hat Enterprise Linux version 6.5 image available from Red Hat[12]. The image has the basic required components installed to deploy a Heat stack. Using Disk Image Builder is an optional step but it allows applications stacks to be deployed rapidly. Also, in cases where a single image is used to deploy many VMs image preparation is a more efficient use of network bandwidth rather than installing each virtual machine separately over the network. Keeping a library of updated images also

12 In addition to installing via yum, the guest image is also available from the Red Hat Customer Portal, https://access.redhat.com/site/downloads/

ensures a secure starting point for all deployments.

Disk Image Builder defines an image build primarily by a list of elements provided. In this reference architecture elements *vm* and *rhel* are used to prepare Red Hat Enterprise Linux virtual machine images. For a list of elements packaged with DIB see */usr/share/diskimage-builder/elements/*.

Custom elements may be created. For this environment OpenShift Enterprise elements have been included in the OpenShift Enterprise Heat template, *openshift-enterprise-broker* and *openshift-enterprise-node*. To include custom elements the custom element path must be specified as a colon-delimited list in the *ELEMENTS_PATH* environment variable.

Many elements use environment variables as parameters. Most elements include a README file explaining the purpose and options for the element.

The steps to preparing an image for deployment:

1. Install diskimage-builder
2. Install the Red Hat Enterprise Linux guest image
3. Execute **`disk-image-create`** with the appropriate arguments
4. Import into OpenStack Glance image service

Images are prepped by executing **`disk-image-create`** with this format.

```
disk-image-create -a <IMAGE_ARCH> -o <OUTPUT_IMAGE_NAME> <ELEMENTS LIST>
```

Since several environment variables are required a script is prepared to simplify the execution of **`disk-image-create`**. Refer to the DIB *README* file as well as relevant element *README* files for available environment variables.

Create and save the following in the file *ose_dib_env.sh*.

```
#!/bin/bash

# assumes rhel-guest-image-6 is installed

# image in GB
export DIB_IMAGE_SIZE=10
export DIB_RHSM_USER=admin
export DIB_RHSM_PASSWORD=BADPASS
export DIB_REG_TYPE=rhn
export DIB_SAT_URL=http://REDACTED/XMLRPC
export DIB_SAT_CERT_RPM_URL=http://REDACTED/pub/rhn-org-trusted-ssl-cert-
1.0-1.noarch.rpm
export DIB_CLOUD_IMAGES=file:////usr/share/rhel-guest-image-6
export BASE_IMAGE_FILE=`ls -1 /usr/share/rhel-guest-image-6/ | grep qcow`
export TMP_BUILD_DIR=$HOME/tmp
export DIB_SAT_KEY=1-ose2-0
export DIB_OSE_VERSION=2.0
export OSE_VERSION=2.0
export DIB_YUM_VALIDATOR_VERSION=2.0
export ELEMENTS_PATH=/usr/share/openstack-heat-templates/openshift-
enterprise/dib/elements/
```

From the OpenStack controller node, prepare the images.

1. Install diskimage-builder and Heat templates. For this reference architecture custom DIB elements are included in openstack-heat-templates so these are installed at this time.

   ```
   # yum install -y diskimage-builder openstack-heat-templates
   ```

2. Install the Red Hat Enterprise Linux guest image. This installs a 322MB qcow2 image in */usr/share/rhel-guest-image-6/*

   ```
   # ls -1 /usr/share/rhel-guest-image-6/
   rhel-guest-image-6-6.5-20140121.0-1
   rhel-guest-image-6-6.5-20140121.0-1.qcow2
   rhel-guest-image-6-6.5-20140121.0-1.xml
   ```

3. Create a *tmp* working directory for DIB.

   ```
   # mkdir /root/tmp
   ```

4. Source the prepared file to export environment variables.

   ```
   # source ose_dib_env.sh
   ```

5. Prepare the images by running **disk-image-create**. The tool must be run twice, once to create the broker image and again to create the node image. Since the Red Hat Enterprise Linux guest image is a very small image each run may take over ten minutes, depending on the network performance of the package source, Red Hat Network or Satellite. The OpenShift Enterprise node image installation is much longer than the broker image. Output truncated below.

   ```
   disk-image-create \
   -a amd64 \
   -o RHEL65-x86_64-broker \
   -p openshift-enterprise-release vm rhel openshift-enterprise-broker |&
   tee /tmp/dib-create-broker.log
   Building elements: base  vm rhel openshift-enterprise-broker
   Expanded element dependencies to: rpm-distro dib-run-parts vm openshift-
   enterprise-broker base rhel openshift-enterprise-repos cache-url
   Building in /root/tmp/image.dzM7MH9x
   dib-run-parts Thu Mar 6 10:43:33 EST 2014 Running
   /root/tmp/image.dzM7MH9x/hooks/root.d/01-ccache
   dib-run-parts Thu Mar 6 10:43:33 EST 2014 01-ccache completed
   dib-run-parts Thu Mar 6 10:43:33 EST 2014 Running
   /root/tmp/image.dzM7MH9x/hooks/root.d/10-rhel-cloud-image
   Fetching Base Image
     % Total    % Received % Xferd  Average Speed   Time    Time     Time
   Current
                                    Dload  Upload   Total   Spent    Left
   Speed
   100  322M  100  322M    0     0   854M      0 --:--:-- --:--:-- --:--:--
   854M
   Server copy has changed. Using server version of
   file:////usr/share/rhel-guest-image-6//rhel-guest-image-6-6.5-
   20140121.0-1.qcow2
   Repacking base image as tarball.
   [output truncated]
   ...
   ```

```
# disk-image-create \
-a amd64 \
-o RHEL65-x86_64-node \
-p openshift-enterprise-release vm rhel openshift-enterprise-node |& tee
/tmp/dib-create-node.log
[output truncated]
...
```

6. Import the prepared images into the Glance image service so they are available for deployment.

```
# glance image-create \
--name RHEL65-OSE-broker \
--is-public true \
--disk-format qcow2 \
--container-format bare \
--file RHEL65-x86_64-broker.qcow2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | 2303ccbc463a1337759347c595aa063b     |
| container_format | bare                                 |
| created_at       | 2014-03-20T01:58:35                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | ca640bbf-4a88-401a-8649-8445334e980e |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | RHEL65-OSE-broker                    |
| owner            | 8ac0aa9d7a284221aac7a9729e0846ce     |
| protected        | False                                |
| size             | 672980480                            |
| status           | active                               |
| updated_at       | 2014-03-20T01:58:41                  |
+------------------+--------------------------------------+
# glance image-create \
--name RHEL65-OSE-node \
--is-public true \
--disk-format qcow2 \
--container-format bare \
--file RHEL65-x86_64-node.qcow2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | f56007bab6e8d5de3fb48e729ecd727f     |
| container_format | bare                                 |
| created_at       | 2014-03-20T01:59:37                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | 11bcddff-fbe6-48c5-9b01-fcb535ccced4 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | RHEL65-OSE-node                      |
| owner            | 8ac0aa9d7a284221aac7a9729e0846ce     |
| protected        | False                                |
| size             | 1559355904                           |
| status           | active                               |
| updated_at       | 2014-03-20T01:59:51                  |
+------------------+--------------------------------------+
```

Prepare RHC Client Image

In this environment a client image is also created using Disk Image Builder. While this is not necessary, it ensures the client is updated and additional packages are included when deployed. If more customization is desired custom elements may be created. For additional information on creating custom elements in Disk Image Builder see the README file[13].

1. On the OpenStack controller host, create the client image, installing the rhc package. Other packages to assist application development and system administration are also included.

```
# disk-image-create \
-a amd64 \
-o RHEL65-x86_64-client \
-p rhc,vim,elinks,bind-utils \
vm rhel |& tee /tmp/dib-create-rhc-client.log
[output truncated]
```

2. Import image into Glance.

```
# glance image-create \
--name RHEL65-OSE-client \
--is-public true \
--disk-format qcow2 \
--container-format bare \
--file RHEL65-x86_64-client.qcow2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | a2eb937350afb65bf907647a8b53f391     |
| container_format | bare                                 |
| created_at       | 2014-03-20T02:01:34                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | f14873fa-c859-4691-be70-86d5c88c4324 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | RHEL65-OSE-client                    |
| owner            | 8ac0aa9d7a284221aac7a9729e0846ce     |
| protected        | False                                |
| size             | 337641472                            |
| status           | active                               |
| updated_at       | 2014-03-20T02:01:36                  |
+------------------+--------------------------------------+
```

13 https://github.com/openstack/diskimage-builder

All of the images necessary for this environment are now available for creating instances.

4.2 OpenShift Enterprise

4.2.1 Heat Template

The deployment of the complex OpenShift Enterprise environment described in this reference architecture has historically involved many manual configuration steps. The Heat template developed for this reference architecture has greatly reduced the time to deploy and configure an HA environment. **Figure 4.2.1.1: Heat Template** depicts the core Heat stack deployment.

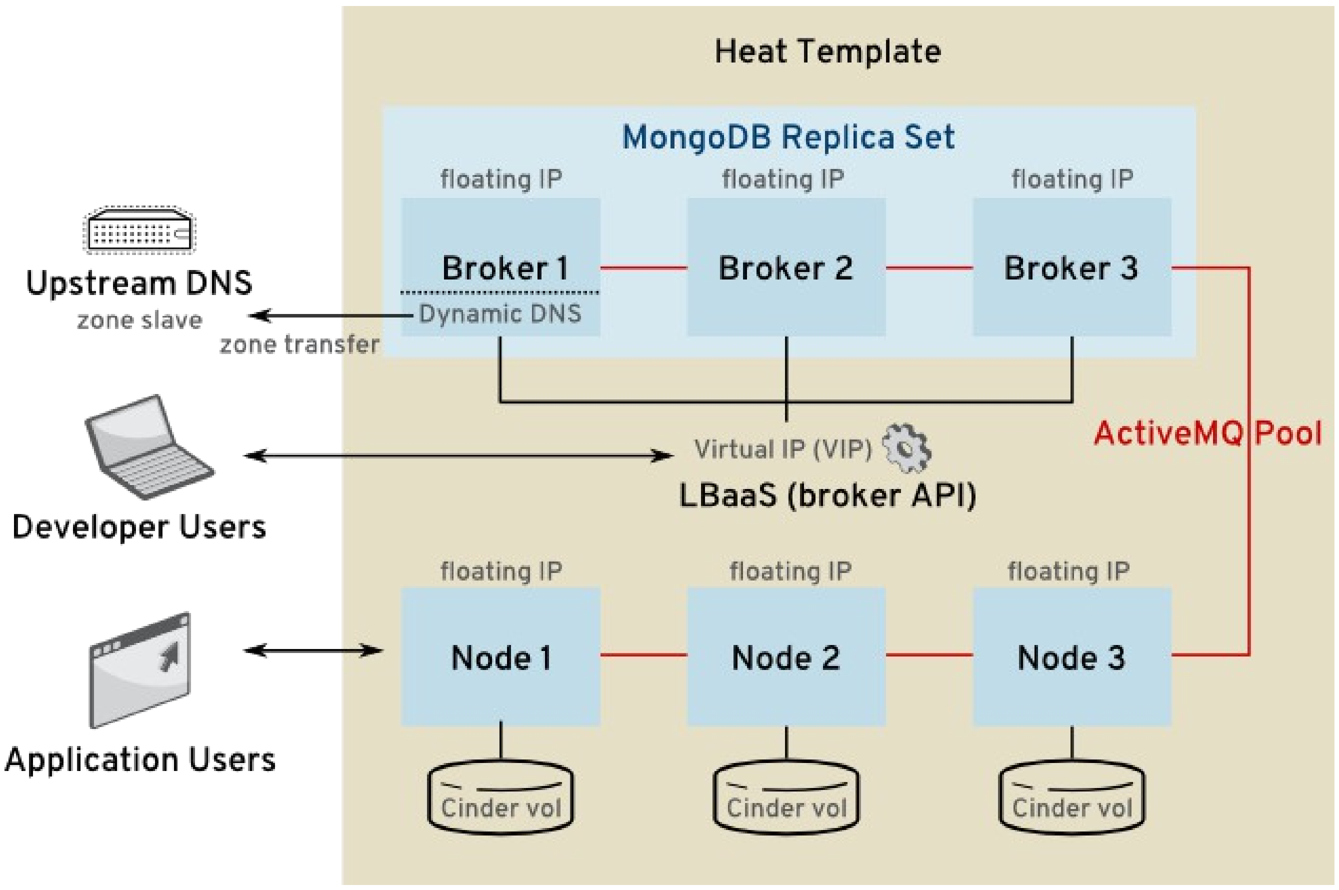


Figure 4.2.1.1: Heat Template

As application load increases additional nodes may be deployed via the Heat template. The node Heat template deploys a single node server with attached cinder volume and a floating IP address. Section **5.6 Adding Nodes** outlines deployment steps.

Heat templates must be installed on the host that deploys Heat stacks. In this environment the Heat client host is the OpenStack controller node. The package **`openstack-heat-templates`** was installed in the previous Disk Image Builder workflow.

List the template files for this reference architecture.

```
# ls -1 /usr/share/openstack-heat-templates/openshift-
enterprise/heat/neutron/highly-available/
ose_ha_env.yaml
ose_ha_stack.yaml
ose_node_env.yaml
ose_node_stack.yaml
README.md
```

Note: Heat templates may change frequently. Expected changes include parameter updates, OpenShift installation script and updates to track changes in the Heat API. Since documentation is limited to this paper and the template *README* file it may be necessary to become familiar with the template itself to understand usage.

4.2.1.1 Parameters

The basis for the OpenShift configuration is the **`openshift.sh`** installer script[14]. Effort has been made in the development of the Heat template to support many of the configuration features of this script. The parameters passed into the script drive the behavior of the configuration. Therefore, where there is a question on what a particular parameter does it is best to refer to the installer script, where the first several hundred lines provide documentation.

14 https://github.com/openshift/openshift-extras/blob/enterprise-2.0/enterprise/install-scripts/generic/openshift.sh

See **Table 4.2.1.1: OpenShift Installer Parameters** for a list of parameters from **`openshift.sh`** supported by the Heat template. A description is provided for each parameter in the Heat template.

Heat Parameter	Installer Parameter	Description
install_method	CONF_INSTALL_METHOD	Installation method
rh_reg_user	CONF_RHN_USER	Red Hat registration user
rh_reg_pass	CONF_RHN_PASS	Red Hat registration password
rh_reg_pool	CONF_SM_REG_POOL	Red Hat registration pool
rh_reg_act_key	CONF_RHN_REG_ACTKEY	Red Hat registration activation key
rh_reg_opts	CONF_RHN_REG_OPTS	Red Hat registration options
domain	CONF_DOMAIN	Domain
hosts_domain	CONF_HOSTS_DOMAIN	Hosts domain
broker{1..3}_hostname	CONF_BROKER_HOSTNAME	Broker 1-3 hostnames
node{1..3}_hostname	CONF_NODE_HOSTNAME	Node 1-3 hostnames
named_hostname	CONF_NAMED_HOSTNAME	DNS server hostname
named_ip	CONF_NAMED_IP_ADDR	DNS server IP address
replicants	CONF_ACTIVEMQ_REPLICANTS CONF_DATASTORE_REPLICANTS	List of AMQ and Mongo replica hosts.
cartridges	CONF_CARTRIDGES	Node cartridges installed
openshift_user1	CONF_OPENSHIFT_USER1	Create OpenShift user
openshift_pass1	CONF_OPENSHIFT_PASSWORD1	OpenShift password
mongo_broker_user	CONF_MONGODB_BROKER_USER	MongoDB broker user
mongo_broker_pass	CONF_MONGODB_BROKER_PASSWORD	MongoDB broker password
mcollective_user	CONF_MCOLLECTIVE_USER	Mcollective user
mcollective_pass	CONF_MCOLLECTIVE_PASSWORD	Mcollective password
activemq_admin_pass	CONF_ACTIVEMQ_ADMIN_PASSWORD	AMQ admin password
activemq_user_pass	CONF_ACTIVEMQ_AMQ_USER_PASSWORD	AMQ user password

Table 4.2.1.1: OpenShift Installer Parameters

In some cases a parameter was not exposed for user configuration. In these cases a default value was chosen for this specific installation environment. See **Table 4.2.1.2: Heat Template-specific Parameters** for a list of parameters unique to the Heat template.

Heat Parameter	Description
load_bal_hostname	Load balancer hostname for brokers
key_name	Key for SSH access
broker_image	Glance image for broker
node_image	Glance image for node
broker_server_flavor	Broker instance flavor
node_server_flavor	Node instance flavor
primary_avail_zone	Primary availability zone
secondary_avail_zone	Secondary availability zone
node_vol_size	Cinder volume size (GB) for node storage
upstream_dns_ip	Upstream DNS server IP
public_net_id	Neutron public network ID
private_net_id	Neutron private network ID
private_subnet_id	Neutron private subnet ID

Table 4.2.1.2: Heat Template-specific Parameters

4.2.1.2 Resources

Heat resources are declared in Heat templates. Therefore Heat templates look more like a data structure than a script. Many resource types are defined in the Heat template guide[15]. Note that Heat templates support many OpenStack resources as well as AWS and other cloud provider resource types. In the context of OpenStack AWS resources are mapped to the analogous OpenStack resources. For example, *AWS::EC2::SecurityGroup* refers to an OpenStack security group.

In the simplified example Heat template below there are two sections: *parameters* and *resources*. In the *resources* section two resources are declared to create a cinder volume and attach the volume to an instance. The *node_vol_size* parameter allows a user to specify the size of the cinder volume but a default is provided if unspecified. Parameters are referenced using the *get_param* function[16]. The *node_vol_size* parameter is provided as a property to the *node_vol* resource.

```
parameters:
  node_vol_size:
    description: Node cinder volume size (GB)
    type: number
    default: 12
```

15 http://docs.openstack.org/developer/heat/template_guide/
16 http://docs.openstack.org/developer/heat/template_guide/hot_spec.html#get-param

```
...
resources:
  node_vol:
    type: OS::Cinder::Volume
    properties:
      name: node_volume
      description: persistent storage volume for node
      size: { get_param: node_vol_size }
  node_attach_vol:
    type: OS::Cinder::VolumeAttachment
    properties:
      instance_uuid: { get_resource: node_instance }
      mountpoint: "/dev/vdb"
      volume_id: { get_resource: node_vol }
```

Resources may be referenced using the *get_resource* function[17]. Resources are created in an arbitrary order as dependent resources are available. In the above example the cinder volume attachment resource will not be created until dependent resources *node_instance* and *node_volume* are created.

See **Table 4.2.1.3: Heat Template Resources** for a list of Heat template resources used in this reference architecture.

Category	Type	Description
Nova	OS::Nova::Server	Nova compute instance
Cinder	OS::Cinder::Volume	Cinder volume
	OS::Cinder::VolumeAttachment	Instance and mountpoint
Neutron networking	OS::Neutron::Port	Neutron port
	OS::Neutron::FloatingIP	Neutron floating IP address
	OS::Neutron::FloatingIPAssociation	Associate floating IP with LBaaS VIP
Neutron LBaaS	OS::Neutron::LoadBalancer	LBaaS
	OS::Neutron::HealthMonitor	LBaaS retries, delay, timeout
	OS::Neutron::Pool	LBaaS pool
Miscellaneous	AWS::EC2::SecurityGroup	Firewall rules
	AWS::CloudFormation::WaitCondition	Control timing of resource creation
	AWS::CloudFormation::WaitConditionHandle	Resource handle called by **`cfn-signal`** for WaitCondition

Table 4.2.1.3: Heat Template Resources

17 http://docs.openstack.org/developer/heat/template_guide/hot_spec.html#get-resource

4.2.1.3 Cloud-init user_data

Heat is primarily concerned with the provisioning of resources. Configuration of those resources is provided by a *cloud-init* script. For many Heat templates there is a shell script embedded into the *user_data* property of *OS::Nova::Server* compute resources[18]. The *user_data* property is where all of the configuration of the OpenShift Environment occurs. In summary the script exports many environment variables from the Heat parameters, downloads and executes the **`openshift.sh`** installer and reboots.

Broker1 Host

On the broker1 server some additional configuration is performed. The *CONF_INSTALL_COMPONENTS* variable includes "*named*" in the list of components so that a dynamic DNS server is configured. This includes installing BIND, creating a master zone file, creating a key and starting **`named`**. The zone file is populated with "A" records for all servers deployed in the stack.

Also on broker1 the *CONF_ACTIONS* variable includes values to configure the MongoDB replica set: "*configure_datastore_add_replicants*".

Node Hosts

The node server *user_data* script contains a large section to configure OpenShift to use the attached volume. The volume must be configured from scratch, including configuring two partitions, filesystems and mount points. Select directories are then moved to the volume. Refer to **Appendix D: Node Attached Storage Script** or the *ose_node_stack.yaml* template for the script.

> **Note:** Due to a bug[19] in the Heat templates, *cloud-init* metadata is blocked by security group egress rules. Egress rules must be commented out from both *ose_ha_stack.yaml* and *ose_node_stack.yaml* files for the user_data to run. Before launching the Heat stack edit these files by commenting out or removing the **boldface** lines below.

ose_ha_stack.yaml

```
resources:
  ose_broker_sec_grp:
    type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: broker firewall rules
      #VpcId: { get_param: private_net_id }
      SecurityGroupIngress:
      - {IpProtocol: tcp, FromPort: '22', ToPort: '22', CidrIp: 0.0.0.0/0}
      - {IpProtocol: udp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '80', ToPort: '80', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '443', ToPort: '443', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '27017', ToPort: '27017', CidrIp: 
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '61613', ToPort: '61613', CidrIp: 
```

18 In a future release of Heat it is expected that configuration will become a separate, shared resource.
19 https://bugzilla.redhat.com/show_bug.cgi?id=1080638

```
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '61616', ToPort: '61616', CidrIp: 
0.0.0.0/0}
      #SecurityGroupEgress:
      #- {IpProtocol: tcp, FromPort: '22', ToPort: '22', CidrIp: 0.0.0.0/0}
      #- {IpProtocol: udp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '27017', ToPort: '27017', CidrIp: 
0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '61613', ToPort: '61613', CidrIp: 
0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '61616', ToPort: '61616', CidrIp: 
0.0.0.0/0}
```

ose_node_stack.yaml

```
resources:
  ose_node_sec_grp:
    type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Node firewall rules
      #VpcId: { get_param: private_net_id }
      SecurityGroupIngress:
      - {IpProtocol: tcp, FromPort: '22', ToPort: '22', CidrIp: 0.0.0.0/0}
      - {IpProtocol: udp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '80', ToPort: '80', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '443', ToPort: '443', CidrIp: 0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '8000', ToPort: '8000', CidrIp: 
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '8443', ToPort: '8443', CidrIp: 
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '2303', ToPort: '2308', CidrIp: 
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '35531', ToPort: '65535', CidrIp: 
0.0.0.0/0}
      - {IpProtocol: tcp, FromPort: '27017', ToPort: '27017', CidrIp: 
0.0.0.0/0}
      #SecurityGroupEgress:
      #- {IpProtocol: udp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '53', ToPort: '53', CidrIp: 0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '443', ToPort: '443', CidrIp: 
0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '35531', ToPort: '65535', CidrIp: 
0.0.0.0/0}
      #- {IpProtocol: tcp, FromPort: '61613', ToPort: '61613', CidrIp: 
0.0.0.0/0}
```

4.2.2 Launch Heat Stack

Heat stacks may be deployed using the **`heat`** CLI tool, which is an interface for the Heat API. In this reference architecture Heat CLI commands are run from the OpenStack *controller* node.

From the *controller* node, execute the following commands.

1. Navigate to the template directory.

```
# cd /usr/share/openstack-heat-templates/openshift-
enterprise/heat/neutron/highly-available/
```

2. Edit the environment template for the current environment, updating the **boldface** values to match the environment.

   The environment file requires neutron network IDs are provided. Use **`neutron net-list`** to print these IDs. The required values are in **boldface** type.

```
# neutron net-list
+--------------------------------------+---------
+------------------------------------------------------+
| id                                   | name    | subnets
|
+--------------------------------------+---------
+------------------------------------------------------+
| d985ff64-06a4-44a7-9197-9711679544e4 | private | b430045c-1337-456f-9f3b-
6c6b45c4e0e5 10.0.6.0/24    |
| e1dc8068-77eb-47d8-8873-8849a816ebd2 | public  | c1841ece-a9af-4579-9f5b-
61c15ff70ce0 10.16.136.0/21 |
```

ose_ha_env.yaml

```
parameters:
  key_name: rootkp
  domain: ose.example.com
  hosts_domain: ose.example.com
  replicants:
broker1.ose.example.com,broker2.ose.example.com,broker3.ose.example.com
  upstream_dns_ip: 10.16.143.247
  node_image: RHEL65-OSE-node
  broker_image: RHEL65-OSE-broker
  broker_server_flavor: m3.medium
  node_server_flavor: m3.medium
  node_vol_size: 12
  activemq_admin_pass: BADPASS
  activemq_user_pass: BADPASS
  mcollective_pass: BADPASS
  mongo_broker_pass: BADPASS
  openshift_pass1: BADPASS
  install_method: rhn
  rh_reg_user: admin
  rh_reg_pass: BADPASS
  rh_reg_act_key: 1-ose2-0
  rh_reg_opts: --serverUrl=http://REDACTED/XMLRPC
  primary_avail_zone: rack1
  secondary_avail_zone: rack2
  cartridges: all
  private_net_id: d985ff64-06a4-44a7-9197-9711679544e4
  public_net_id: e1dc8068-77eb-47d8-8873-8849a816ebd2
  private_subnet_id: b430045c-1337-456f-9f3b-6c6b45c4e0e5
resource_registry:
```

```
OpenShift::Node::Server: ose_node_stack.yaml
```

3. Launch the Heat stack.

```
# heat stack-create ose-ha -f ose_ha_stack.yaml -e ose_ha_env.yaml
+--------------------------------------+------------+--------------------
+----------------------+
| id                                   | stack_name | stack_status       |
creation_time         |
+--------------------------------------+------------+--------------------
+----------------------+
| 3742039c-3cf8-442c-8a1b-11609a2e606b | ose-ha     | CREATE_IN_PROGRESS | 2014-03-
26T15:55:50Z |
+--------------------------------------+------------+--------------------
+----------------------+
```

The Heat stack is created in the background. There are several ways to monitor the creation of stack resources.

- Periodically check list the Heat stack

```
# heat stack-list
+--------------------------------------+------------+--------------------
+----------------------+
| id                                   | stack_name | stack_status       |
creation_time         |
+--------------------------------------+------------+--------------------
+----------------------+
| 0563be5c-3f2c-45c2-b400-e0d20ab9c32a | ose-ha     | CREATE_IN_PROGRESS | 2014-03-
26T16:11:35Z |
+--------------------------------------+------------+--------------------
+----------------------+
```

- Check the Heat resource list.

```
# heat resource-list ose-ha
+-----------------------+------------------------------------------+--------------------
+----------------------+
| resource_name         | resource_type                            | resource_status    | updated_time
|
+-----------------------+------------------------------------------+--------------------
+----------------------+
| broker1_wait_handle   | AWS::CloudFormation::WaitConditionHandle | CREATE_COMPLETE    | 2014-03-
26T15:55:52Z |
| broker_wait_handle    | AWS::CloudFormation::WaitConditionHandle | CREATE_COMPLETE    | 2014-03-
26T15:55:52Z |
| monitor               | OS::Neutron::HealthMonitor               | CREATE_COMPLETE    | 2014-03-
26T15:55:52Z |
| ose_broker_sec_grp    | AWS::EC2::SecurityGroup                  | CREATE_COMPLETE    | 2014-03-
26T15:55:52Z |
| broker1_wait_condition | AWS::CloudFormation::WaitCondition      | CREATE_IN_PROGRESS | 2014-03-
26T15:55:53Z |
| broker_wait_condition  | AWS::CloudFormation::WaitCondition      | CREATE_IN_PROGRESS | 2014-03-
26T15:55:53Z |
| broker1_port          | OS::Neutron::Port                        | CREATE_COMPLETE    | 2014-03-
26T15:55:54Z |
| broker2_port          | OS::Neutron::Port                        | CREATE_COMPLETE    | 2014-03-
26T15:55:54Z |
| broker3_port          | OS::Neutron::Port                        | CREATE_COMPLETE    | 2014-03-
26T15:55:54Z |
| lb_vip_port           | OS::Neutron::Port                        | CREATE_COMPLETE    | 2014-03-
26T15:55:54Z |
| pool                  | OS::Neutron::Pool                        | CREATE_COMPLETE    | 2014-03-
26T15:55:54Z |
| broker1_floating_ip   | OS::Neutron::FloatingIP                  | CREATE_COMPLETE    | 2014-03-
26T15:55:56Z |
| broker2_floating_ip   | OS::Neutron::FloatingIP                  | CREATE_COMPLETE    | 2014-03-
26T15:55:56Z |
| broker3_floating_ip   | OS::Neutron::FloatingIP                  | CREATE_COMPLETE    | 2014-03-
```

```
26T15:55:56Z |
| lb_vip_floating_ip     | OS::Neutron::FloatingIP                  | CREATE_COMPLETE    | 2014-03-
26T15:55:56Z |
| mylb                   | OS::Neutron::LoadBalancer                | CREATE_COMPLETE    | 2014-03-
26T15:55:56Z |
| broker2_instance       | OS::Nova::Server                         | CREATE_IN_PROGRESS | 2014-03-
26T15:55:57Z |
| broker3_instance       | OS::Nova::Server                         | CREATE_IN_PROGRESS | 2014-03-
26T15:55:58Z |
| node1_instance         | OpenShift::Node::Server                  | CREATE_IN_PROGRESS | 2014-03-
26T15:55:59Z |
| node3_instance         | OpenShift::Node::Server                  | CREATE_IN_PROGRESS | 2014-03-
26T15:56:01Z |
| node2_instance         | OpenShift::Node::Server                  | CREATE_IN_PROGRESS | 2014-03-
26T15:56:03Z |
| lb_pool_vip            | OS::Neutron::FloatingIPAssociation       | CREATE_COMPLETE    | 2014-03-
26T15:56:06Z |
| broker1_instance       | OS::Nova::Server                         | INIT_COMPLETE      | 2014-03-
26T15:56:38Z |
+------------------------+------------------------------------------+--------------------
+----------------------+
```

- Tail heat-engine.log (output truncated)

```
# tail -f /var/log/heat/heat-engine.log
2014-03-26 11:56:03.638 8100 INFO heat.engine.resource [-] Validating CinderVolumeAttachment
"node_attach_vol"
2014-03-26 11:56:03.663 8100 INFO heat.engine.resource [-] creating SecurityGroup
"ose_node_sec_grp"
2014-03-26 11:56:04.652 8100 INFO heat.engine.resource [-] creating CinderVolume "node_vol"
2014-03-26 11:56:04.671 8100 INFO urllib3.connectionpool [-] Starting new HTTP connection (1):
10.16.139.90
2014-03-26 11:56:06.049 8100 INFO urllib3.connectionpool [-] Starting new HTTP connection (1):
10.16.139.90
2014-03-26 11:56:06.198 8100 INFO heat.engine.resource [-] creating Port "node_port"
2014-03-26 11:56:06.769 8100 INFO urllib3.connectionpool [-] Starting new HTTP connection (1):
10.16.139.90
2014-03-26 11:56:06.875 8100 INFO heat.engine.resource [-] creating Port "node_port"
2014-03-26 11:56:07.085 8100 INFO urllib3.connectionpool [-] Starting new HTTP connection (1):
10.16.139.90
2014-03-26 11:56:07.226 8100 INFO heat.engine.resource [-] creating Port "node_port"
2014-03-26 11:56:08.585 8100 INFO heat.engine.resource [-] creating FloatingIP
"node_floating_ip"
2014-03-26 11:56:08.892 8100 INFO urllib3.connectionpool [-] Starting new HTTP connection (1):
10.16.139.90
2014-03-26 11:56:09.016 8100 INFO heat.engine.resource [-] creating FloatingIP
"node_floating_ip"
2014-03-26 11:56:09.245 8100 INFO heat.engine.resource [-] creating FloatingIP
"node_floating_ip"
2014-03-26 11:56:10.608 8100 INFO heat.engine.resource [-] creating Server "node_instance"
2014-03-26 11:56:11.519 8100 INFO heat.engine.resource [-] creating Server "node_instance"
2014-03-26 11:56:12.027 8100 INFO heat.engine.resource [-] creating Server "node_instance"
2014-03-26 11:56:13.982 8100 INFO heat.engine.environment [-] Registering
OpenShift::Node::Server -> file:///usr/share/openstack-heat-templates/openshift-
enterprise/heat/neutron/highly-available/ose_node_stack.yaml
```

View list of instances created with associated floating IP addresses.

```
# nova list --fields name,status,networks
+--------------------------------------+-------------+--------+----------------------------------+
| ID                                   | Name        | Status | Networks                         |
+--------------------------------------+-------------+--------+----------------------------------+
| 131f6d56-1326-44cf-adde-62fece707117 | ose_broker1 | ACTIVE | private=10.0.6.6, 10.16.138.130  |
| 63f62c45-a897-4bb1-8985-115effa8a86e | ose_broker2 | ACTIVE | private=10.0.6.2, 10.16.138.132  |
| b395fa99-c9c3-47dc-a1ce-4e351a021745 | ose_broker3 | ACTIVE | private=10.0.6.7, 10.16.138.129  |
| 4c78d5af-b17c-48ca-aa64-26ad9dff20ae | ose_node    | ACTIVE | private=10.0.6.9, 10.16.138.134  |
| 8ac3828c-4d80-4c18-a568-7d239c15f942 | ose_node    | ACTIVE | private=10.0.6.8, 10.16.138.133  |
| ce7eaef3-6623-43f3-801f-3bbae5278f69 | ose_node    | ACTIVE | private=10.0.6.10, 10.16.138.135 |
+--------------------------------------+-------------+--------+----------------------------------+
```

The OpenStack *Horizon* web user interface (webUI) may be used as an alternative to the

`heat` CLI workflow. This may be preferred in cases where the Heat template presents a large number of optional and required parameters.

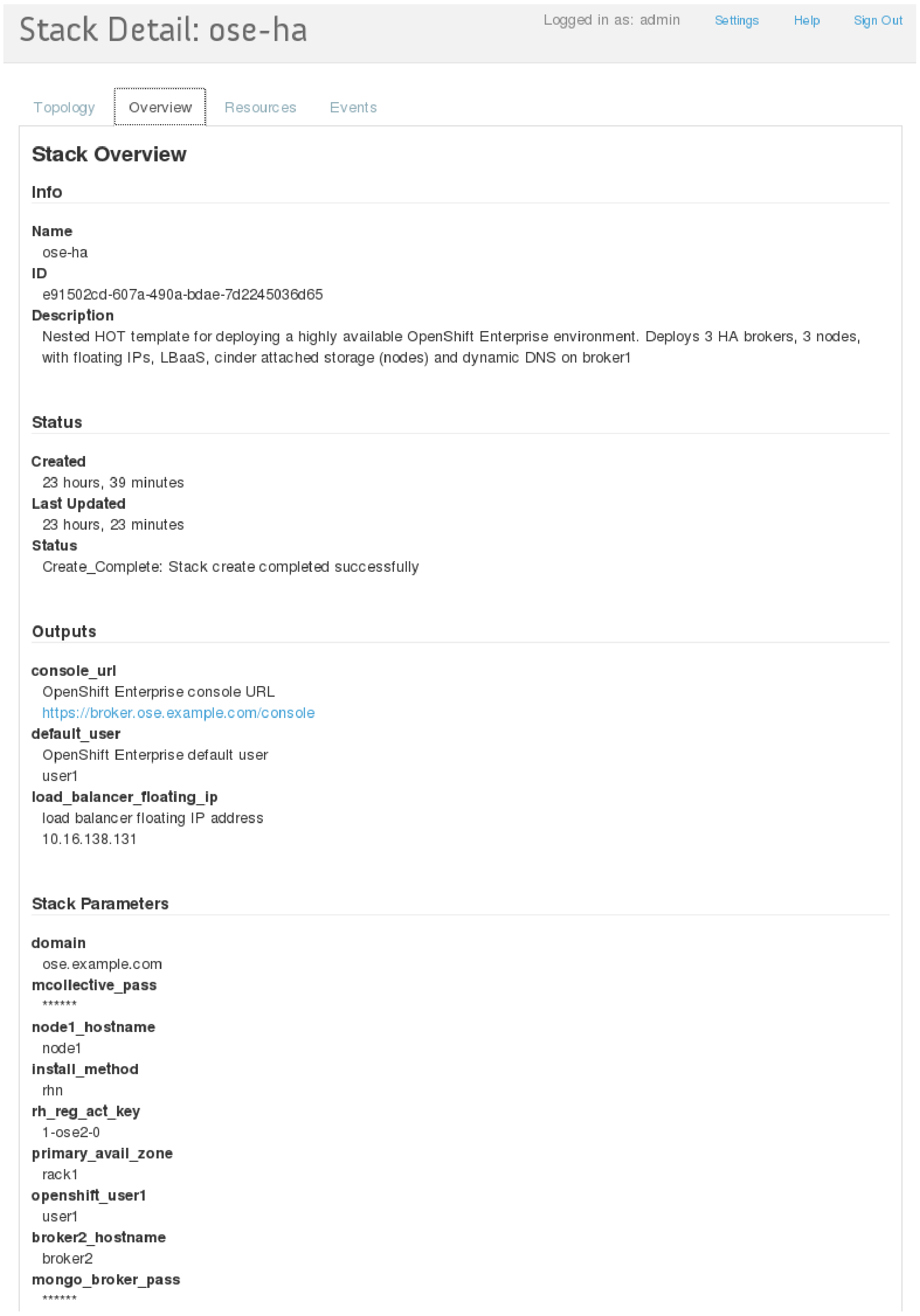
> **Note**: Horizon does not currently support environment files and nested templates so it is not compatible with this workflow.

In this demonstration the webUI is used to view the output of the completed stack. Heat stack ouput is useful for understanding the configuration deployed.

Using a web browser load the URL of the OpenStack controller node.

1. Navigate to **Stacks**
2. Click **OpenShift** stack
3. Click **Overview** tab

See **Figure 4.2.2.1: Heat stack output detail** for a cropped screenshot of the Heat stack detail.


Stack Detail: ose-ha

Stack Overview

Info

Name
ose-ha
ID
e91502cd-607a-490a-bdae-7d2245036d65
Description
Nested HOT template for deploying a highly available OpenShift Enterprise environment. Deploys 3 HA brokers, 3 nodes, with floating IPs, LBaaS, cinder attached storage (nodes) and dynamic DNS on broker1

Status

Created
23 hours, 39 minutes
Last Updated
23 hours, 23 minutes
Status
Create_Complete: Stack create completed successfully

Outputs

console_url
OpenShift Enterprise console URL
https://broker.ose.example.com/console
default_user
OpenShift Enterprise default user
user1
load_balancer_floating_ip
load balancer floating IP address
10.16.138.131

Stack Parameters

domain
ose.example.com
mcollective_pass
******
node1_hostname
node1
install_method
rhn
rh_reg_act_key
1-ose2-0
primary_avail_zone
rack1
openshift_user1
user1
broker2_hostname
broker2
mongo_broker_pass
******

Figure 4.2.2.1: Heat stack output detail

SSH

Once the Heat stack is deployed VMs may be access via SSH using the nova key specified for the Heat stack. Disk Image Builder and Heat create an admin user account *ec2-user* by default. By default is no *root* user password defined on the Red Hat guest image.

```
# ssh -i rootkp.pem ec2-user@10.16.138.130
```

For root access change to root.

```
[ec2-user@broker1 ~]$ sudo -i
[root@broker1 ~]#
```

Note, if an attempt to login using the root user, instructions are provided:

```
# ssh -i rootkp.pem root@10.16.138.130
Warning: Permanently added '10.16.138.130' (RSA) to the list of known hosts.
Please login as the user "ec2-user" rather than the user "root".

ctrl+c
^CConnection to 10.16.138.130 closed.
```

4.2.3 Configure DNS

Dynamic DNS updates are authenticated using a BIND key. The BIND key on the dynamic DNS server on broker1 must be provided to the OpenShift DNS plugin for successful dynamic updates. The Heat deployment does not insert the BIND key on broker2 and broker3. The *BIND_KEYVALUE* must be copied from broker1 to broker2 and broker3.

1. On broker1 view the contents of the DNS key file for the zone. Copy the BIND key value.

```
# cat /var/named/ose.example.com.key
key ose.example.com {
  algorithm "HMAC-MD5";
  secret
"5bt8CRo7u6d5NVsE+xz+oVwTavWHwnmoVdTo7QfDgA20Ks71iYcDoisYnAwu7tHVrnrqM8D
IwSbLLhFGGT3uwA==";
};
```

2. On broker2, edit */etc/openshift/plugins.d/openshift-origin-dns-nsupdate.conf* and paste the BIND key value from broker1 as the value for the *BIND_KEYVALUE*.

```
BIND_SERVER="10.16.138.130"
BIND_PORT=53
BIND_ZONE="ose.example.com"
BIND_KEYNAME="ose.example.com"
BIND_KEYVALUE="5bt8CRo7u6d5NVsE+xz+oVwTavWHwnmoVdTo7QfDgA20Ks71iYcDoisYn
Awu7tHVrnrqM8DIwSbLLhFGGT3uwA=="
BIND_KEYALGORITHM="HMAC-MD5"
```

3. Restart broker services

```
# service openshift-broker restart
Stopping openshift-broker:                                 [  OK  ]
Starting openshift-broker: httpd: Could not reliably determine the
server's fully qualified domain name, using broker2.ose.example.com for
ServerName
                                                           [  OK  ]
```

Note: The warning that the FQDN could not be determined is due to httpd requiring reverse lookup of the hostname. For this environment reverse hostname lookup is not configured.

4. Repeat for broker3.

In this reference architecture an “upstream” DNS server refers to an existing DNS server that is the OpenShift zone slave. While DNS configuration is likely very different in an enterprise setting, these steps are provided as an example configuration from the lab.

Add slave zone to upstream DNS server.

1. On the upstream DNS server add the new zone to */etc/named.conf*. This can be appended to the end of the file.

```
# cat > /etc/named.conf << EOF
zone "ose.example.com" {
      type slave;
      file "slaves/ose.example.com.db";
      masters { 10.16.138.130; };
};
EOF
```

2. Restart named on the upstream DNS server.

```
# service named restart
Stopping named: .                                          [  OK  ]
Starting named:                                            [  OK  ]
```

The Heat template populates the broker and node hostnames in DNS. The load balanced hostname *broker* is also mapped to the LBaaS IP address for broker access. This allows developer users to access the broker API via *broker.ose.example.com*. The node hosts in the environment are also pointed to the load balanced hostname.

4.2.4 Rename Nodes

Since the same Heat template is used for each node in this nested Heat stack, nova labels all of the node instances as “ose-node”. As an optional step for convenience it may be desired to rename the instances.

1. On the OpenStack controller lookup the IP address via hostname.

```
# host node1.ose.example.com
node1.ose.example.com has address 10.16.138.134
```

2. List instances.

```
# nova list --fields name,networks
+--------------------------------------+-------------+---------------------------------+
| ID                                   | Name        | Networks                        |
+--------------------------------------+-------------+---------------------------------+
| 131f6d56-1326-44cf-adde-62fece707117 | ose_broker1 | private=10.0.6.6, 10.16.138.130  |
| 63f62c45-a897-4bb1-8985-115effa8a86e | ose_broker2 | private=10.0.6.2, 10.16.138.132  |
| b395fa99-c9c3-47dc-a1ce-4e351a021745 | ose_broker3 | private=10.0.6.7, 10.16.138.129  |
| 4c78d5af-b17c-48ca-aa64-26ad9dff20ae | ose_node    | private=10.0.6.9, 10.16.138.134  |
| 8ac3828c-4d80-4c18-a568-7d239c15f942 | ose_node    | private=10.0.6.8, 10.16.138.133  |
| ce7eaef3-6623-43f3-801f-3bbae5278f69 | ose_node    | private=10.0.6.10, 10.16.138.135 |
+--------------------------------------+-------------+---------------------------------+
```

3. Note the instance ID with address 10.16.138.136 and rename.

```
# nova rename 4c78d5af-b17c-48ca-aa64-26ad9dff20ae ose-node1
```

4. Repeat above steps for all node instances, node2 and node3. List the renamed instances.

```
# nova list --fields name,networks
+--------------------------------------+-------------+---------------------------------+
| ID                                   | Name        | Networks                        |
+--------------------------------------+-------------+---------------------------------+
| 4c78d5af-b17c-48ca-aa64-26ad9dff20ae | ose-node1   | private=10.0.6.9, 10.16.138.134 |
| ce7eaef3-6623-43f3-801f-3bbae5278f69 | ose-node2   | private=10.0.6.10, 10.16.138.135 |
| 8ac3828c-4d80-4c18-a568-7d239c15f942 | ose-node3   | private=10.0.6.8, 10.16.138.133 |
| 131f6d56-1326-44cf-adde-62fece707117 | ose_broker1 | private=10.0.6.6, 10.16.138.130 |
| 63f62c45-a897-4bb1-8985-115effa8a86e | ose_broker2 | private=10.0.6.2, 10.16.138.132 |
| b395fa99-c9c3-47dc-a1ce-4e351a021745 | ose_broker3 | private=10.0.6.7, 10.16.138.129 |
+--------------------------------------+-------------+---------------------------------+
```

4.2.5 Configure LBaaS

The Heat template provides most of the LBaaS configuration. Members must be added using the **`neutron lb-member-create`** command. The private, fixed IP address is used from **`nova list`** output.

```
# nova list --fields name,networks
+--------------------------------------+-------------+---------------------------------+
| ID                                   | Name        | Networks                        |
+--------------------------------------+-------------+---------------------------------+
| 4c78d5af-b17c-48ca-aa64-26ad9dff20ae | ose-node1   | private=10.0.6.9, 10.16.138.134 |
| ce7eaef3-6623-43f3-801f-3bbae5278f69 | ose-node2   | private=10.0.6.10, 10.16.138.135 |
| 8ac3828c-4d80-4c18-a568-7d239c15f942 | ose-node3   | private=10.0.6.8, 10.16.138.133 |
| 131f6d56-1326-44cf-adde-62fece707117 | ose_broker1 | private=10.0.6.6, 10.16.138.130 |
| 63f62c45-a897-4bb1-8985-115effa8a86e | ose_broker2 | private=10.0.6.2, 10.16.138.132 |
| b395fa99-c9c3-47dc-a1ce-4e351a021745 | ose_broker3 | private=10.0.6.7, 10.16.138.129 |
+--------------------------------------+-------------+---------------------------------+
# neutron lb-member-create --address 10.0.6.6 --protocol-port 443
ose_broker_lb_pool
Created a new member:
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| address            | 10.0.6.6                             |
| admin_state_up     | True                                 |
| id                 | 319e0147-e80a-43c7-98e5-c41551f339f4 |
| pool_id            | f000e0f4-34a6-4da7-8443-d6afd079d5e4 |
| protocol_port      | 443                                  |
| status             | PENDING_CREATE                       |
| status_description |                                      |
| tenant_id          | 68d6039cfa0c4395bda913ef432167da     |
| weight             | 1                                    |
+--------------------+--------------------------------------+
# neutron lb-member-create --address 10.0.6.2 --protocol-port 443
ose_broker_lb_pool
Created a new member:
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| address            | 10.0.6.2                             |
| admin_state_up     | True                                 |
| id                 | 6405a90b-bb4c-4763-9e95-d3cc3d6455a3 |
| pool_id            | f000e0f4-34a6-4da7-8443-d6afd079d5e4 |
| protocol_port      | 443                                  |
| status             | PENDING_CREATE                       |
| status_description |                                      |
| tenant_id          | 68d6039cfa0c4395bda913ef432167da     |
| weight             | 1                                    |
+--------------------+--------------------------------------+
# neutron lb-member-create --address 10.0.6.7 --protocol-port 443
ose_broker_lb_pool
Created a new member:
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| address            | 10.0.6.7                             |
| admin_state_up     | True                                 |
| id                 | e9b8a600-98a8-4f84-aa90-b5b5a8c43d6d |
| pool_id            | f000e0f4-34a6-4da7-8443-d6afd079d5e4 |
| protocol_port      | 443                                  |
| status             | PENDING_CREATE                       |
| status_description |                                      |
| tenant_id          | 68d6039cfa0c4395bda913ef432167da     |
| weight             | 1                                    |
+--------------------+--------------------------------------+
# neutron lb-pool-show ose_broker_lb_pool
+------------------------+--------------------------------------------------------+
| Field                  | Value                                                  |
+------------------------+--------------------------------------------------------+
| admin_state_up         | True                                                   |
| description            | Load balancer for OpenShift Enterprise broker hosts    |
| health_monitors        | ef7e0e15-2226-4107-95f4-2929c3fea731                   |
| health_monitors_status | {"monitor_id": "ef7e0e15-2226-4107-95f4-2929c3fea731", |
|                        |  "status": "ACTIVE", "status_description": null}       |
| id                     | f000e0f4-34a6-4da7-8443-d6afd079d5e4                   |
| lb_method              | ROUND_ROBIN                                            |
| members                | 319e0147-e80a-43c7-98e5-c41551f339f4                   |
|                        | 6405a90b-bb4c-4763-9e95-d3cc3d6455a3                   |
|                        | e9b8a600-98a8-4f84-aa90-b5b5a8c43d6d                   |
| name                   | ose_broker_lb_pool                                     |
| protocol               | HTTPS                                                  |
| provider               | haproxy                                                |
| status                 | ACTIVE                                                 |
| status_description     |                                                        |
| subnet_id              | b430045c-1337-456f-9f3b-6c6b45c4e0e5                   |
| tenant_id              | 68d6039cfa0c4395bda913ef432167da                       |
| vip_id                 | d9743eab-6f89-4284-9b8f-e597cf8849bd                   |
+------------------------+--------------------------------------------------------+
```

The Heat template also does not configure session persistence. Session persistence is important with secure connections. Without it each call to the server may be served from a different server and will need to establish a new session. In this reference the persistence method is IP address. Persistence based on session cookie is also supported.

1. List the load balanced VIP.

```
# neutron lb-vip-list
+--------------------------------------+----------------+----------+----------
+----------------+--------+
| id                                   | name           | address  | protocol |
admin_state_up | status |
+--------------------------------------+----------------+----------+----------
+----------------+--------+
| d9743eab-6f89-4284-9b8f-e597cf8849bd | ose_broker_vip | 10.0.6.4 | HTTPS    | True
| ACTIVE |
+--------------------------------------+----------------+----------+----------
+----------------+--------+
```

2. Using the VIP name from step 1, show the VIP details. Notice session persistence is not defined.

```
# neutron lb-vip-show ose_broker_vip
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| address            | 10.0.6.4                             |
| admin_state_up     | True                                 |
| connection_limit   | -1                                   |
| description        | broker virtual IP (VIP)              |
| id                 | d9743eab-6f89-4284-9b8f-e597cf8849bd |
| name               | ose_broker_vip                       |
| pool_id            | f000e0f4-34a6-4da7-8443-d6afd079d5e4 |
| port_id            | 72476a98-6588-4eb1-be1e-f7b79d7ca430 |
| protocol           | HTTPS                                |
| protocol_port      | 443                                  |
| status             | ACTIVE                               |
| status_description |                                      |
| subnet_id          | b430045c-1337-456f-9f3b-6c6b45c4e0e5 |
| tenant_id          | 68d6039cfa0c4395bda913ef432167da     |
+--------------------+--------------------------------------+
```

3. Update the VIP to add session persistence, type *SOURCE_IP*.

```
# neutron lb-vip-update ose_broker_vip --session-persistence type=dict
type='SOURCE_IP'
Updated vip: ose_broker_vip
```

4. Show the VIP details again, noting session_persistence is now defined.

```
# neutron lb-vip-show ose_broker_vip
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 10.0.6.4                             |
| admin_state_up      | True                                 |
| connection_limit    | -1                                   |
| description         | broker virtual IP (VIP)              |
| id                  | d9743eab-6f89-4284-9b8f-e597cf8849bd |
| name                | ose_broker_vip                       |
| pool_id             | f000e0f4-34a6-4da7-8443-d6afd079d5e4 |
| port_id             | 72476a98-6588-4eb1-be1e-f7b79d7ca430 |
| protocol            | HTTPS                                |
| protocol_port       | 443                                  |
| session_persistence | {"type": "SOURCE_IP"}                |
| status              | ACTIVE                               |
| status_description  |                                      |
| subnet_id           | b430045c-1337-456f-9f3b-6c6b45c4e0e5 |
| tenant_id           | 68d6039cfa0c4395bda913ef432167da     |
+---------------------+--------------------------------------+
```

4.2.6 Synchronizing Keys and Certificates

In a distributed, multi-host environment it is important to synchronize certain keys and certificates to improve the use and management of the OpenShift Enterprise environment.

Copy rsync Public Key

The broker **rsync** public key needs to be authorized on nodes to enable moving gears. The install script puts a copy of the public key on the broker web server as a convenience for this purpose to enable copying the key manually. Run this command from all OpenShift node hosts.

```
# for i in {1..3}; do  curl -k https://broker$i/rsync_id_rsa.pub >>
/root/.ssh/authorized_keys; done
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
102   410  102   410    0     0   4064      0 --:--:-- --:--:-- --:--:--  200k
```

Copy ssh and httpd Key and Certificate

To avoid errors when gears are moved all node hosts should have the same **ssh** keys and *https* key and certificate. This will also ensure **ssh** and **git** don't return spurious warnings about the host keys changing.

1. From broker1, copy the files from node1 into a temporary directory.

```
$ sudo -i
# mkdir keys
# cd keys
# scp -i /root/.ssh/rsync_id_rsa \
root@node1.ose.example.com:/etc/pki/tls/private/localhost.key .
localhost.key                 100% 1704     1.7KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa
root@node1.ose.example.com:/etc/pki/tls/certs/localhost.crt .
localhost.crt                 100% 1521     1.5KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa \
root@node1.ose.example.com:/etc/ssh/ssh_* .
ssh_config                    100% 2047     2.0KB/s   00:00
ssh_host_dsa_key              100%  672     0.7KB/s   00:00
ssh_host_dsa_key.pub          100%  590     0.6KB/s   00:00
ssh_host_key                  100%  963     0.9KB/s   00:00
ssh_host_key.pub              100%  627     0.6KB/s   00:00
ssh_host_rsa_key              100% 1675     1.6KB/s   00:00
ssh_host_rsa_key.pub          100%  382     0.4KB/s   00:00
```

2. Copy the files from broker1 to node2 and node3.

```
# scp -i /root/.ssh/rsync_id_rsa localhost.key \
root@node2.ose.example.com:/etc/pki/tls/private/.
The authenticity of host 'node2.ose.example.com (10.16.138.135)' can't
be established.
RSA key fingerprint is bf:ff:9e:8b:0a:db:16:37:b2:b5:de:9e:09:d6:2d:1a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'node2.ose.example.com,10.16.138.135' (RSA)
to the list of known hosts.
localhost.key                 100% 1704     1.7KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa localhost.key \
root@node3.ose.example.com:/etc/pki/tls/private/.
The authenticity of host 'node3.ose.example.com (10.16.138.133)' can't
be established.
RSA key fingerprint is f9:f0:6b:84:db:65:7f:ba:47:28:2a:2f:f0:23:cf:c6.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'node3.ose.example.com,10.16.138.133' (RSA)
to the list of known hosts.
localhost.key                100% 1704     1.7KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa localhost.crt \
root@node2.ose.example.com:/etc/pki/tls/certs/.
localhost.crt                100% 1521     1.5KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa localhost.crt \
root@node3.ose.example.com:/etc/pki/tls/certs/.
localhost.crt                100% 1521     1.5KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa ssh_* \
root@node2.ose.example.com:/etc/ssh/.
ssh_config                   100% 2047     2.0KB/s   00:00
ssh_host_dsa_key             100%  672     0.7KB/s   00:00
ssh_host_dsa_key.pub         100%  590     0.6KB/s   00:00
ssh_host_key                 100%  963     0.9KB/s   00:00
ssh_host_key.pub             100%  627     0.6KB/s   00:00
ssh_host_rsa_key             100% 1675     1.6KB/s   00:00
ssh_host_rsa_key.pub         100%  382     0.4KB/s   00:00
# scp -i /root/.ssh/rsync_id_rsa ssh_* \
root@node3.ose.example.com:/etc/ssh/.
ssh_config                   100% 2047     2.0KB/s   00:00
ssh_host_dsa_key             100%  672     0.7KB/s   00:00
ssh_host_dsa_key.pub         100%  590     0.6KB/s   00:00
ssh_host_key                 100%  963     0.9KB/s   00:00
ssh_host_key.pub             100%  627     0.6KB/s   00:00
ssh_host_rsa_key             100% 1675     1.6KB/s   00:00
ssh_host_rsa_key.pub         100%  382     0.4KB/s   00:00
```

3. Reboot node2 and node3.

4.2.7 Test Configuration

There are several tools included with OpenShift Enterprise to verify the configuration.

1. On a broker host check that the node hosts are communicating via Mcollective as part of the ActiveMQ pool.

   ```
   # oo-mco ping
   node1.ose.example.com                    time=97.44 ms
   node3.ose.example.com                    time=137.24 ms
   node2.ose.example.com                    time=137.77 ms


   ---- ping statistics ----
   3 replies max: 137.77 min: 97.44 avg: 124.15
   ```

   Note: In some cases during the testing of this environment the node hosts required a reboot before joining the ActiveMQ pool.

2. On a broker host run the diagnostics tool. Use the *-v* argument for verbose output.

   ```
   # oo-diagnostics
   ```

```
WARN: test_node_profiles_districts_from_broker
        No districts are defined. Districts should be used in any
         production installation. Please consult the Administration
         Guide.

WARN: test_altered_package_owned_configs
        The mlocate package is not installed. mlocate is not a
         required runtime package; however, you may install mlocate
         to enable further diagnostics checking.

WARN: test_broker_certificate
Using a self-signed certificate for the broker
3 WARNINGS
NO ERRORS
```

Output shows three expected warnings. One is a reminder to create districts for applications. The other is to install the **`mlocate`** tool for package checking. Install this tool if desired.

```
# yum install -y mlocate
```

The last warning is a regarding using a self-signed certificate. In an enterprise installation the organization's signed SSL certificate from a third party certificate authority should be installed.

3. On a node host run the node diagnostics tool. Use the *-v* argument for verbose output.

```
# oo-accept-node
PASS
```

5 Operational Management

5.1 Manage Districts and Profiles

A node district defines a set of node hosts. A gear profile defines the size of a gear. Districts and profiles are used together to organize and control placement of applications. It is important to assign each node host to a district to enable gear migration.

In this section a medium gear size and medium profile is added to the environment. Gear high availability (HA) is also enabled.

Perform the following steps on all broker hosts.

1. Enable HA.

```
# sed -i \
's/ALLOW_HA_APPLICATIONS="false"/ALLOW_HA_APPLICATIONS="true"/'
/etc/openshift/broker.conf
```

2. Add medium gear size.

```
# sed -i \
's/VALID_GEAR_SIZES="small"/VALID_GEAR_SIZES="small,medium"/' \
/etc/openshift/broker.conf
```

3. Allow medium gear size to be available to users by default.

```
# set -i \
's/DEFAULT_GEAR_CAPABILITIES="small"/DEFAULT_GEAR_CAPABILITIES="small,me
dium"/' /etc/openshift/broker.conf
```

4. Restart openshift-broker to enable the changes.

```
# service openshift-broker restart
Stopping openshift-broker:                                 [  OK  ]
Starting openshift-broker: httpd: Could not reliably determine the
server's fully qualified domain name, using broker1.ose.example.com for
ServerName
                                                           [  OK  ]
```

A *small* profile is configured by default on each node. To enable another profile on a particular node edit the *node_profile* parameter in */etc/openshift/resource_limits.conf* on node3 and restart Mcollective. This will ensure *small* profile applications are placed on node1 and node2 and *medium* profile applications will be placed on node3.

Perform this command on node3 only.

```
# sed -i 's/node_profile=small/node_profile=medium/'
/etc/openshift/resource_limits.conf
# service ruby193-mcollective restart
Shutting down mcollective:                                 [  OK  ]
Starting mcollective:                                      [  OK  ]
```

Districts may now be defined. Perform the following on any broker host.

1. Create small and medium districts associated with the small and medium profiles.

```
# oo-admin-ctl-district -c create -n small_district -p small
Successfully created district: 53331d33e5c3748173000001

{"_id"=>"53331d33e5c3748173000001",
 "uuid"=>"53331d33e5c3748173000001",
 "available_uids"=>"<6000 uids hidden>",
 "name"=>"small_district",
 "gear_size"=>"small",
 "server_identities"=>[],
 "available_capacity"=>6000,
 "max_uid"=>6999,
 "max_capacity"=>6000,
 "active_server_identities_size"=>0,
 "updated_at"=>2014-03-26 18:32:19 UTC,
 "created_at"=>2014-03-26 18:32:19 UTC}
# oo-admin-ctl-district -c create -n medium_district -p medium
Successfully created district: 53331d3fe5c374a54d000001

{"_id"=>"53331d3fe5c374a54d000001",
 "uuid"=>"53331d3fe5c374a54d000001",
 "available_uids"=>"<6000 uids hidden>",
 "name"=>"medium_district",
 "gear_size"=>"medium",
 "server_identities"=>[],
 "available_capacity"=>6000,
 "max_uid"=>6999,
 "max_capacity"=>6000,
 "active_server_identities_size"=>0,
 "updated_at"=>2014-03-26 18:32:31 UTC,
 "created_at"=>2014-03-26 18:32:31 UTC}
```

2. Add node1 and node2 to the *small_district*.

```
# oo-admin-ctl-district -c add-node -n small_district -i
node1.ose.example.com
Success!

{"_id"=>"53331d33e5c3748173000001",
 "active_server_identities_size"=>1,
 "available_capacity"=>6000,
 "available_uids"=>"<6000 uids hidden>",
 "created_at"=>2014-03-26 18:32:19 UTC,
 "gear_size"=>"small",
 "max_capacity"=>6000,
 "max_uid"=>6999,
 "name"=>"small_district",
 "server_identities"=>[{"name"=>"node1.ose.example.com",
"active"=>true}],
 "updated_at"=>2014-03-26 18:36:32 UTC,
 "uuid"=>"53331d33e5c3748173000001"}
# oo-admin-ctl-district -c add-node -n small_district -i
node2.ose.example.com
Success!

{"_id"=>"53331d33e5c3748173000001",
 "active_server_identities_size"=>2,
 "available_capacity"=>6000,
 "available_uids"=>"<6000 uids hidden>",
 "created_at"=>2014-03-26 18:32:19 UTC,
 "gear_size"=>"small",
 "max_capacity"=>6000,
 "max_uid"=>6999,
 "name"=>"small_district",
 "server_identities"=>
  [{"name"=>"node1.ose.example.com", "active"=>true},
   {"name"=>"node2.ose.example.com", "active"=>true}],
 "updated_at"=>2014-03-26 18:41:20 UTC,
 "uuid"=>"53331d33e5c3748173000001"}
```

3. Add node3 to the *medium_district*.

```
# oo-admin-ctl-district -c add-node -n medium_district -i
node3.ose.example.com
Success!

{"_id"=>"53331d3fe5c374a54d000001",
 "active_server_identities_size"=>1,
 "available_capacity"=>6000,
 "available_uids"=>"<6000 uids hidden>",
 "created_at"=>2014-03-26 18:32:31 UTC,
 "gear_size"=>"medium",
 "max_capacity"=>6000,
 "max_uid"=>6999,
 "name"=>"medium_district",
 "server_identities"=>[{"name"=>"node3.ose.example.com",
"active"=>true}],
 "updated_at"=>2014-03-26 18:42:55 UTC,
 "uuid"=>"53331d3fe5c374a54d000001"}
```

5.2 Users

OpenShift Enterprise authenticates users through the remote-user authentication plugin which relies on the httpd service to pass on the authenticated user. In this reference architecture OpenShift Enterprise users are defined locally on each broker using Apache Basic Authentication. Local user *user1* was defined on each broker during deployment configuration. This user is for demonstration purposes only.

In an enterprise setting the plugin would be configured for LDAP or Kerberos authentication for user management. For an in-depth discussion on user authentication options refer to the pending reference architecture "OpenShift 2.0 Integration with Red Hat Identity Management"[20]. Otherwise see section *Configuring User Authentication* in the OpenShift Enterprise Deployment Guide[21].

New users are allowed to the small and medium gear sizes configured in section **5.1 Manage Districts and Profiles**. Existing users must have new gear sizes added manually. User management may be performed on any broker.

20 https://access.redhat.com/search/browse/articles
21 https://access.redhat.com/site/documentation/en-US/OpenShift_Enterprise/2/html-single/Deployment_Guide/index.html#sect-Configuring_User_Authentication

```
# oo-admin-ctl-user -l user1 --addgearsize medium


Adding gear size medium for user user1... Done.

User user1:
                            plan:
                consumed domains: 0
                     max domains: 10
                  consumed gears: 0
                       max gears: 100
    max tracked storage per gear: 0
  max untracked storage per gear: 0
                      gear sizes: small, medium
            sub accounts allowed: false
private SSL certificates allowed: false
              inherit gear sizes: false
                      HA allowed: false
```

HA applications have been allowed but the capability must be enabled per user.

```
# oo-admin-ctl-user -l user1 --allowha true


Setting HA capability to true for user user1... Done.

User user1:
                            plan:
                consumed domains: 0
                     max domains: 10
                  consumed gears: 0
                       max gears: 100
    max tracked storage per gear: 0
  max untracked storage per gear: 0
                      gear sizes: small, medium
            sub accounts allowed: false
private SSL certificates allowed: false
              inherit gear sizes: false
                      HA allowed: true
```

5.3 Install Client

In this reference architecture a separate client instance is deployed and configured for creating and managing OpenShift Enterprise applications. The OpenShift client guest image prepared in section **4.1.5 Image Preparation with Disk Image Builder** will be used for the client instance.

5.3.1 Launch Client Instance

1. List the Glance images created previously. Note the client image (bold).

```
# glance image-list --human-readable
+--------------------------------------+------------------+-------------+------------------
+---------+--------+
| ID                                   | Name             | Disk Format | Container Format |
Size    | Status |
+--------------------------------------+------------------+-------------+------------------
+---------+--------+
| ca640bbf-4a88-401a-8649-8445334e980e | RHEL65-OSE-broker | qcow2       | bare             |
641.8MB | active |
| f14873fa-c859-4691-be70-86d5c88c4324 | RHEL65-OSE-client | qcow2       | bare             |
322MB   | active |
| 11bcddff-fbe6-48c5-9b01-fcb535ccced4 | RHEL65-OSE-node   | qcow2       | bare             |
1.5GB   | active |
+--------------------------------------+------------------+-------------+------------------
+---------+--------+
```

2. List the network IDs. Note the **boldface** private network ID.

```
# neutron net-list
+--------------------------------------+---------
+-----------------------------------------------------+
| id                                   | name    | subnets
|
+--------------------------------------+---------
+-----------------------------------------------------+
| d985ff64-06a4-44a7-9197-9711679544e4 | private | b430045c-1337-456f-9f3b-6c6b45c4e0e5
10.0.6.0/24    |
| e1dc8068-77eb-47d8-8873-8849a816ebd2 | public  | c1841ece-a9af-4579-9f5b-61c15ff70ce0
10.16.136.0/21 |
+--------------------------------------+---------
+-----------------------------------------------------+
```

3. Boot the client instance.

```
# nova boot --flavor 2 --image RHEL65-OSE-client --nic net-id=d30a1652-
3485-47ad-9a0a-b1aa5cebd74a --key-name rootkp rhc-client
+--------------------------------------+--------------------------------------+
| Property                             | Value                                |
+--------------------------------------+--------------------------------------+
| OS-EXT-STS:task_state                | scheduling                           |
| image                                | RHEL65-OSE-client                    |
| OS-EXT-STS:vm_state                  | building                             |
| OS-EXT-SRV-ATTR:instance_name        | instance-0000000c                    |
| OS-SRV-USG:launched_at               | None                                 |
| flavor                               | m1.small                             |
| id                                   | 94c5a533-2e77-4d26-ba17-6553b5910819 |
| security_groups                      | [{u'name': u'default'}]              |
| user_id                              | 1bbb071cbdec4fd8aaec33dc804b0102     |
| 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                              | 2014-03-26T18:56:21Z                 |
| hostId                               |                                      |
| OS-EXT-SRV-ATTR:host                 | None                                 |
| OS-SRV-USG:terminated_at             | None                                 |
| key_name                             | rootkp                               |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | None                                 |
| name                                 | rhc-client                           |
| adminPass                            | 9NcTrFKSaLNZ                         |
| tenant_id                            | 68d6039cfa0c4395bda913ef432167da     |
| created                              | 2014-03-26T18:56:20Z                 |
| os-extended-volumes:volumes_attached | []                                   |
| metadata                             | {}                                   |
+--------------------------------------------+--------------------------------------------+
```

4. Creating a floating IP address using the instance ID and the instance port ID.

```
# nova list --name rhc-client --fields name,networks
+--------------------------------------+------------+-------------------+
| ID                                   | Name       | Networks          |
+--------------------------------------+------------+-------------------+
| 94c5a533-2e77-4d26-ba17-6553b5910819 | rhc-client | private=10.0.6.11 |
+--------------------------------------+------------+-------------------+
# neutron port-list --device_id 94c5a533-2e77-4d26-ba17-6553b5910819
+--------------------------------------+------+-------------------
+-------------------------------------------------------------------------------------+
| id                                   | name | mac_address       | fixed_ips
|
+--------------------------------------+------+-------------------
+-------------------------------------------------------------------------------------+
| 4b154d08-a4a8-4500-b804-851c0030c016 |      | fa:16:3e:ea:0a:5e | {"subnet_id": "b430045c-
1337-456f-9f3b-6c6b45c4e0e5", "ip_address": "10.0.6.11"} |
+--------------------------------------+------+-------------------
+-------------------------------------------------------------------------------------+
# neutron floatingip-create --port-id 4b154d08-a4a8-4500-b804-
851c0030c016 public
Created a new floatingip:
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| fixed_ip_address    | 10.0.6.11                            |
| floating_ip_address | 10.16.138.136                        |
| floating_network_id | e1dc8068-77eb-47d8-8873-8849a816ebd2 |
| id                  | 5220d568-a540-4d5e-afbf-58f9aa081bef |
| port_id             | 4b154d08-a4a8-4500-b804-851c0030c016 |
| router_id           | de662e52-fe2a-4709-9f22-6cc438d5c48a |
| tenant_id           | 68d6039cfa0c4395bda913ef432167da     |
+---------------------+--------------------------------------+
# nova list --name rhc-client --fields name,networks
+--------------------------------------+------------+----------------------------------+
| ID                                   | Name       | Networks                         |
+--------------------------------------+------------+----------------------------------+
| 94c5a533-2e77-4d26-ba17-6553b5910819 | rhc-client | private=10.0.6.11, 10.16.138.136 |
+--------------------------------------+------------+----------------------------------+
```

The rhc client VM is now available.

5.3.2 Configure Client Tools

Once the instance has booted the **rhc** client tools may be configured. The default *cloud-user* is used to access the VM via SSH.

```
# ssh -i rootkp.pem cloud-user@10.16.138.136

$ rhc setup --server broker.ose.example.com -l user1
OpenShift Client Tools (RHC) Setup Wizard

This wizard will help you upload your SSH keys, set your application
namespace, and check that other programs like Git are properly
installed.

Using user1 to login to broker.ose.example.com
```

```
The server's certificate is self-signed, which means that a secure
connection can't be established to 'broker.ose.example.com'.

You may bypass this check, but any data you send to the server could be
intercepted by others.

Connect without checking the certificate? (yes|no): yes
Password: ********

OpenShift can create and store a token on disk which allows to you to access
the server without using your password. The key is stored in
your home directory and should be kept secret.  You can delete the key at
any time by running 'rhc logout'.
Generate a token now? (yes|no) yes
Generating an authorization token for this client ... lasts about 1 day

Saving configuration to /home/cloud-user/.openshift/express.conf ... done

No SSH keys were found. We will generate a pair of keys for you.

    Created: /home/cloud-user/.ssh/id_rsa.pub

Your public SSH key must be uploaded to the OpenShift server to access code.
Upload now? (yes|no) yes

Since you do not have any keys associated with your OpenShift account, your
new key will be uploaded as the 'default' key.

Uploading key 'default' ... done

Checking for git ... found git version 1.7.1

Checking common problems .. done

Checking for a domain ... none

Applications are grouped into domains - each domain has a unique name
(called a namespace) that becomes part of your public application
URL. You may create your first domain here or leave it blank and use 'rhc
create-domain' later. You will not be able to create an
application without completing this step.

Please enter a namespace (letters and numbers only) |<none>|: syseng
Your domain 'syseng' has been successfully created

Checking for applications ... none

Run 'rhc create-app' to create your first application.

  Do-It-Yourself 0.1                         rhc create-app <app name> diy-0.1
  JBoss Enterprise Application Platform 6 rhc create-app <app name>
jbosseap-6
  Jenkins Server                             rhc create-app <app name> jenkins-
1
  Node.js 0.10                               rhc create-app <app name> nodejs-
```

```
0.10
  PHP 5.3                                rhc create-app <app name> php-5.3
  Perl 5.10                              rhc create-app <app name> perl-
5.10
  Python 2.6                             rhc create-app <app name> python-
2.6
  Python 2.7                             rhc create-app <app name> python-
2.7
  Ruby 1.8                               rhc create-app <app name> ruby-1.8
  Ruby 1.9                               rhc create-app <app name> ruby-1.9
  Tomcat 6 (JBoss EWS 1.0)               rhc create-app <app name>
jbossews-1.0
  Tomcat 7 (JBoss EWS 2.0)               rhc create-app <app name>
jbossews-2.0

  You are using 0 of 100 total gears
  The following gear sizes are available to you: small, medium

Your client tools are now configured.
```

5.4 Deploy Applications

Once the client is configured applications may be created from the client VM. First an auto-scaled application will be created using the default gear size. This deploys an application on one of the small profile nodes, node1 or node2.

```
$ rhc create-app rubyscale ruby-1.9 -s
Application Options
-------------------
  Domain:     syseng
  Cartridges: ruby-1.9
  Gear Size:  default
  Scaling:    yes

Creating application 'rubyscale' ... done


Waiting for your DNS name to be available ... done

Initialized empty Git repository in /home/cloud-user/rubyscale/.git/
The authenticity of host 'rubyscale-syseng.ose.example.com (10.16.138.135)'
can't be established.
RSA key fingerprint is bf:ff:9e:8b:0a:db:16:37:b2:b5:de:9e:09:d6:2d:1a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'rubyscale-syseng.ose.example.com,10.16.138.135'
(RSA) to the list of known hosts.

Your application 'rubyscale' is now available.

  URL:        http://rubyscale-syseng.ose.example.com/
  SSH to:     5333262fe5c374e3ed000005@rubyscale-syseng.ose.example.com
  Git remote: ssh://5333262fe5c374e3ed000005@rubyscale-
syseng.ose.example.com/~/git/rubyscale.git/
```

```
  Cloned to:  /home/cloud-user/rubyscale

Run 'rhc show-app rubyscale' for more details about your app.
```

Next an application using the medium gear is deployed. This deploys an application on the medium profile node, node3.

```
$ rhc create-app pythonmedium python-2.7 -g medium
Application Options
-------------------
  Domain:     syseng
  Cartridges: python-2.7
  Gear Size:  medium
  Scaling:    no

Creating application 'pythonmedium' ... done


Waiting for your DNS name to be available ... done

Initialized empty Git repository in /home/cloud-user/pythonmedium/.git/
The authenticity of host 'pythonmedium-syseng.ose.example.com
(10.16.138.133)' can't be established.
RSA key fingerprint is f9:f0:6b:84:db:65:7f:ba:47:28:2a:2f:f0:23:cf:c6.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'pythonmedium-
syseng.ose.example.com,10.16.138.133' (RSA) to the list of known hosts.

Your application 'pythonmedium' is now available.

  URL:        http://pythonmedium-syseng.ose.example.com/
  SSH to:     533326a2e5c374e3ed00002b@pythonmedium-syseng.ose.example.com
  Git remote: ssh://533326a2e5c374e3ed00002b@pythonmedium-
syseng.ose.example.com/~/git/pythonmedium.git/
  Cloned to:  /home/cloud-user/pythonmedium

Run 'rhc show-app pythonmedium' for more details about your app.
```

5.5 Administrator Console

The administrator console was introduced in OpenShift Enterprise version 2.0. It provides an alternative view of the OpenShift Enterprise environment. The console is read-only; No administrative actions may be performed from the console.

The admin console is not enabled by default. In this environment the console is enabled on broker1 only. Edit */etc/httpd/conf.d/000002_openshift_origin_broker_proxy.conf* file and restart **`httpd`** on broker1.

1. Add the **boldface** lines to the VirtualHost section of file */etc/httpd/conf.d/000002_openshift_origin_broker_proxy.conf*.

```
...
<VirtualHost *:443>
```

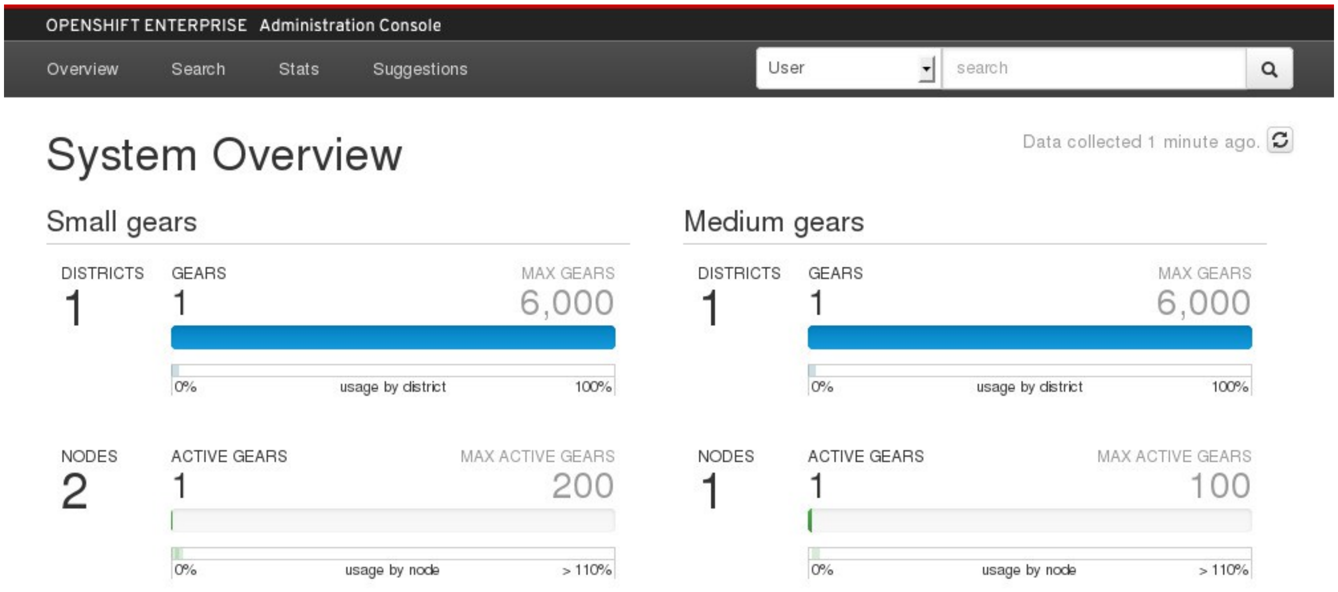
```
  # ServerName we will inherit from other config;
  # ServerAlias is to make sure "localhost" traffic goes here
regardless.
  ServerAlias localhost
  ServerAdmin root@localhost
  DocumentRoot /var/www/html
  RewriteEngine              On
  RewriteRule     ^/$    https://%{HTTP_HOST}/console [R,L]
  SSLEngine on
  SSLProxyEngine on
  SSLCertificateFile /etc/pki/tls/certs/localhost.crt
  SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
  RequestHeader set X_FORWARDED_PROTO 'https'
  RequestHeader set Front-End-Https "On"
  ProxyTimeout 300
  ProxyPass /console http://127.0.0.1:8118/console
  ProxyPassReverse /console http://127.0.0.1:8118/console

  ProxyPass /broker http://127.0.0.1:8080/broker
  ProxyPass /admin-console http://127.0.0.1:8080/admin-console
  ProxyPass /assets http://127.0.0.1:8080/assets
  ProxyPassReverse / http://127.0.0.1:8080/
</VirtualHost>
...
```

2. Restart **httpd**

```
# service httpd restart
Stopping httpd:                                            [  OK  ]
Starting httpd:                                            [  OK  ]
```

Using a browser navigate to https://broker1.ose.example.com/admin-console.



System Overview

Data collected 1 minute ago.

Small gears

DISTRICTS 1 GEARS 1 MAX GEARS 6,000

0% usage by district 100%

NODES 2 ACTIVE GEARS 1 MAX ACTIVE GEARS 200

0% usage by node > 110%

Medium gears

DISTRICTS 1 GEARS 1 MAX GEARS 6,000

0% usage by district 100%

NODES 1 ACTIVE GEARS 1 MAX ACTIVE GEARS 100

0% usage by node > 110%

Figure 5.5.1: OpenShift Enterprise Admin console

If desired the admin console may be enabled on all broker hosts. Perform the above steps on broker2 and broker3. In this case the admin console is accessible from the load balanced hostname https://broker.ose.example.com/admin-console.

5.6 Adding Nodes

For this environment a forth node will be deployed to demonstrate extending the capacity of the OpenShift Enterprise environment. This node will be used to extend the medium district.

The node Heat stack deploys a single node server with attached cinder volume and a floating IP address.

1. Edit the node stack environment file *ose_node_env.yaml* to match the deployment environment.

```
parameters:
  key_name: rootkp
  domain: ose.example.com
  hosts_domain: ose.example.com
  broker1_floating_ip: 10.16.138.130
  node_vol_size: 12
  replicants:
broker1.ose.example.com,broker2.ose.example.com,broker3.ose.example.com
  node_image: RHEL65-OSE-node
  activemq_admin_pass: BADPASS
  activemq_user_pass: BADPASS
  mcollective_pass: BADPASS
  cartridges: all
  install_method: rhn
  rh_reg_user: admin
  rh_reg_pass: BADPASS
  rh_reg_act_key: 1-ose2-0
  rh_reg_opts: --serverUrl=http://REDACTED/XMLRPC
  private_net_id: d985ff64-06a4-44a7-9197-9711679544e4
  public_net_id: e1dc8068-77eb-47d8-8873-8849a816ebd2
  private_subnet_id: b430045c-1337-456f-9f3b-6c6b45c4e0e5
```

2. Launch the stack. Two parameters are overridden: *node_hostname* and *avail_zone* so these nodes have a unique hostname and are placed on the desired OpenStack compute node.

```
# heat stack-create ose_node4 \
-f ose_node_stack.yaml \
-e ose_node_env.yaml \
-P "node_hostname=node4;avail_zone=rack1"
+--------------------------------------+------------+--------------------
+----------------------+
| id                                   | stack_name | stack_status       |
creation_time        |
+--------------------------------------+------------+--------------------
+----------------------+
| 0563be5c-3f2c-45c2-b400-e0d20ab9c32a | ose-ha     | CREATE_COMPLETE    | 2014-03-
26T16:11:35Z |
| b20d8a3e-9f10-41d9-ab7a-d8e51e65e453 | ose_node4  | CREATE_IN_PROGRESS | 2014-03-
26T19:25:37Z |
+--------------------------------------+------------+--------------------
+----------------------+
```

Refer to section **4.2.2 Launch Heat Stack** for monitoring the Heat stack.

List the new instance to determine the floating IP address.

```
# nova list --name ose_node --fields name,networks
+--------------------------------------+----------+----------------------------------+
| ID                                   | Name     | Networks                         |
+--------------------------------------+----------+----------------------------------+
| 8f7fb43f-eca8-48b8-b831-6bc2dbea490f | ose_node | private=10.0.6.12, 10.16.138.137 |
+--------------------------------------+----------+----------------------------------+
```

Once the nodes are deployed the new nodes must be added to the DNS server on broker1 so that node host and IP address resolve. The OpenShift Enterprise **`oo-register-dns`** tool is provided for this use case. Perform the following on one of the broker hosts.

1. Use the **`oo-register-dns`** tool for each node added and restart **named**.

```
# oo-register-dns \
--with-node-hostname node4 \
--with-node-ip 10.16.138.137 \
--domain ose.example.com \
--dns-server broker1.ose.example.com
# service named restart
Stopping named: .                                          [  OK  ]
Starting named:                                            [  OK  ]
```

2. Confirm the zone transfer by querying the new hostnames from **rhc-client**.

```
# dig node4.ose.example.com
; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.23.rc1.el6_5.1 <<>>
node4.ose.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51227
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;node4.ose.example.com.           IN    A

;; ANSWER SECTION:
node4.ose.example.com. 180   IN    A     10.16.138.137

;; AUTHORITY SECTION:
ose.example.com. 1     IN    NS    broker1.ose.example.com.

;; ADDITIONAL SECTION:
broker1.ose.example.com. 1  IN    A     10.16.138.130

;; Query time: 3 msec
;; SERVER: 10.16.143.247#53(10.16.143.247)
;; WHEN: Wed Mar 26 15:46:36 2014
;; MSG SIZE  rcvd: 93
```

5.6.1 Node Configuration

Several configuration steps are required to integrate the added node host to the environment.

Copy the broker rsync public key from the broker hosts. Run this command from node4.

```
# curl -k https://broker1/rsync_id_rsa.pub >> /root/.ssh/authorized_keys
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
102   410  102   410    0     0   4064      0 --:--:-- --:--:-- --:--:--  200k
```

1. From broker1 confirm the node is registered with the environment.

```
# oo-mco ping
node1.ose.example.com                    time=97.58 ms
node3.ose.example.com                    time=137.52 ms
node2.ose.example.com                    time=138.12 ms
node4.ose.example.com                    time=138.65 ms



---- ping statistics ----
4 replies max: 138.65 min: 97.58 avg: 127.97
```

2. From the OpenStack controller node, rename nodes as was done in section **4.2.4 Rename Nodes**. On the OpenStack controller lookup the IP address via hostname. List the instances to discover the instance ID and rename the node.

```
# host node4.ose.example.com
node4.ose.example.com has address 10.16.138.137
# nova list --fields name,networks
+--------------------------------------+------------+--------------------------------+
| ID                                   | Name       | Networks                       |
+--------------------------------------+-------------+---------------------------------+
| 4c78d5af-b17c-48ca-aa64-26ad9dff20ae | ose-node1   | private=10.0.6.9, 10.16.138.134  |
| ce7eaef3-6623-43f3-801f-3bbae5278f69 | ose-node2   | private=10.0.6.10, 10.16.138.135 |
| 8ac3828c-4d80-4c18-a568-7d239c15f942 | ose-node3   | private=10.0.6.8, 10.16.138.133  |
| 131f6d56-1326-44cf-adde-62fece707117 | ose_broker1 | private=10.0.6.6, 10.16.138.130  |
| 63f62c45-a897-4bb1-8985-115effa8a86e | ose_broker2 | private=10.0.6.2, 10.16.138.132  |
| b395fa99-c9c3-47dc-a1ce-4e351a021745 | ose_broker3 | private=10.0.6.7, 10.16.138.129  |
| 8f7fb43f-eca8-48b8-b831-6bc2dbea490f | ose_node    | private=10.0.6.12, 10.16.138.137 |
| 94c5a533-2e77-4d26-ba17-6553b5910819 | rhc-client  | private=10.0.6.11, 10.16.138.136 |
+--------------------------------------+-------------+---------------------------------+
# nova rename 8f7fb43f-eca8-48b8-b831-6bc2dbea490f ose-node4
```

The new node is intended to extend capacity for the *medium_district*. First enable the profile from node4, then add the node to the district from a broker host.

1. Enable the *medium* profile from node4.

```
# sed -i 's/node_profile=small/node_profile=medium/'
/etc/openshift/resource_limits.conf
# service ruby193-mcollective restart
Shutting down mcollective:                                [  OK  ]
Starting mcollective:                                     [  OK  ]
```

2. Add the node to the *medium_district* from any broker host.

```
# oo-admin-ctl-district -c add-node -n medium_district -i
node4.ose.example.com
Success!

{"_id"=>"53331d3fe5c374a54d000001",
 "active_server_identities_size"=>2,
 "available_capacity"=>5999,
 "available_uids"=>"<5999 uids hidden>",
 "created_at"=>2014-03-26 18:32:31 UTC,
 "gear_size"=>"medium",
 "max_capacity"=>6000,
 "max_uid"=>6999,
 "name"=>"medium_district",
 "server_identities"=>
  [{"name"=>"node3.ose.example.com", "active"=>true},
   {"name"=>"node4.ose.example.com", "active"=>true}],
 "updated_at"=>2014-03-26 20:04:23 UTC,
 "uuid"=>"53331d3fe5c374a54d000001"}
```

5.7 Responding to Node Failure

Part of providing a highly available service includes responding to failure. In this section the steps to recovering a failed OpenShift Enterprise node host are described.

The Heat template used to deploy the node hosts moves the necessary files for failed node recovery to an attached storage volume. This allows an organization to integrate existing backup infrastructure to ensure application data is not lost.

1. Create a new node host by following the procedures from section **5.6 Adding Nodes**. To provide the quickest resolution time a hot backup node host may be deployed in advance so it is available for immediate recovering.

2. Detach and delete the Cinder volume created by the node deployment using **`nova volume-detach`** and **`nova volume-delete`** commands. This will be replaced by the Cinder volume in step 3.
3. Attach the Cinder volume from the failed node host using **`nova volume-attach`** commands.
4. Reassign the floating IP from the failed node host to the recovered node host. This may be performed using **`neutron floating-ip-disassociate`** and **`neutron floating-ip-associate`** commands. This will ensure the running applications continue to be served from the same IP address.
5. Reboot the recovered node host.

6 Conclusion

This reference architecture demonstrated how a highly available OpenShift Enterprise 2.0 environment may be rapidly deployed using OpenStack Heat templates. The HOT-based templates developed specifically for this reference demonstrated the capabilities of Heat in automating the orchestration of OpenStack resources needed for a highly available OpenShift Enterprise environment. Procedures for extending and maintaining the environment were also demonstrated.

The procedures demonstrated contain the details necessary to provide Red Hat customers with the ability to reproduce this reference architecture in their own environments.

Appendix A: Revision History

Revision 1.0	March 31, 2014	Aaron Weitekamp
Initial Release		

Appendix B: Contributors

Contributor	Title	Contribution
Keith Schincke	Senior Software Engineer	Review
Mark Heslin	Principle Software Engineer	Review
Steven Dake	Supervisor, Software Engineering	Review
Chris Alphonso	Principle Software Engineer	Review

Appendix C: Logs

Component	Log	Path
OpenShift common	Installer	/tmp/openshift.out
	Heat and cloud-init output	/var/log/heat-provision.log /var/log/cloud-init-output.log
	user_data install script for cloud-init	/var/lib/heat-cfntools/cfn-userdata
	Syslog	/var/log/messages
OpenShift broker	ActiveMQ	/var/log/activemq/*.log
	MongoDB	/var/log/mongodb/mongodb.log
	DNS (broker1 only)	/var/named/data/named.run
	httpd	/var/log/httpd/* /var/www/openshift/broker/log/production.log /var/www/openshift/broker/httpd/logs/* /var/www/openshift/console/log/production.log /var/www/openshift/console/httpd/logs/*
	User action	/var/log/openshift/user_action.log
OpenShift node	Mcollective	/var/log/mcollective.log
OpenStack	LBaaS agent	/var/log/neutron/lbaas-agent.log
	Neutron	/var/log/neutron/server.log
	Heat engine	/var/log/heat/heat-engine.log
	Heat API	/var/log/heat/heat-api.log
	Cinder	/var/log/cinder/*

Table 6.1: Logs

Appendix D: Node Attached Storage Script

This is a section of the user_data property of the *ose_node_stack.yaml* template. It configures attached block storage with two partitions, creates filesystems and moves the OpenShift directories to the mounted partitions.

```
# configure attached storage and move ose dirs
parted --script /dev/vdb -- mklabel msdos
parted --script /dev/vdb -- mkpart primary 1024 5G
parted --script /dev/vdb -- mkpart primary 5G -1s
partx -a /dev/vdb
mkfs.ext4 /dev/vdb1
mkfs.ext4 /dev/vdb2
mkdir /var/lib/node_share
mount /dev/vdb1 /var/lib/node_share
mkdir /var/lib/node_share/etc_openshift
service ruby193-mcollective stop
shopt -s dotglob
mv /etc/openshift/* /var/lib/node_share/etc_openshift
chcon --reference /etc/openshift /var/lib/node_share/etc_openshift
echo "/dev/vdb1 /var/lib/node_share ext4 defaults" >> /etc/fstab
echo "/var/lib/node_share/etc_openshift /etc/openshift none bind,auto" >>
/etc/fstab
mount /etc/openshift
mount /dev/vdb2 /mnt
mv /var/lib/openshift/* /mnt
chcon --reference /var/lib/openshift /mnt
umount /mnt
echo "/dev/vdb2 /var/lib/openshift ext4 usrquota,defaults" >> /etc/fstab
mount /var/lib/openshift
quotacheck --create-files --no-remount --user --group /var/lib/openshift
restorecon /var/lib/openshift/aquota.user
quotaon /var/lib/openshift
service ruby193-mcollective start
reboot
```

Appendix E: Software Versions

The following table lists the relevant software versions used in this environment.

Software Package	Version
diskimage-builder	0.1.5-3
haproxy	1.4.24-2
openstack-utils	2013.2-3
openstack-cinder	2013.2.2-2
openstack-nova-novncproxy	2013.2.2-2
openstack-neutron-openvswitch	2013.2.2-5
openstack-heat-common	2013.2.2-1
python-django-openstack-auth	1.1.2-2
openstack-dashboard	2013.2.2-1
openstack-heat-templates	0-0.3.20140307git
openstack-packstack	2013.2.1-0.28.dev989
openstack-glance	2013.2.2-2
openstack-nova-api	2013.2.2-2
openstack-nova-console	2013.2.2-2
openstack-nova-scheduler	2013.2.2-2
openstack-heat-engine	2013.2.2-1
openstack-heat-api-cfn	2013.2.2-1
openstack-keystone	2013.2.2-1
openstack-nova-common	2013.2.2-2
openstack-nova-conductor	2013.2.2-2
openstack-neutron	2013.2.2-5
openstack-heat-api	2013.2.2-1
openstack-heat-api-cloudwatch	2013.2.2-1
openstack-nova-compute	2013.2.2-2

Table 6.2: Software Versions—Red Hat OpenStack Platform 4.0

Software Package	Version
rhc	1.17.5.2-1
rubygem-openshift-origin-common	1.17.2.7-1
openshift-origin-broker-util	1.17.6.4-1
rubygem-openshift-origin-dns-nsupdate	1.15.2-1
openshift-origin-util-scl	1.16.1-2
openshift-origin-msg-common	1.16.3-1
rubygem-openshift-origin-controller	1.17.12.3-1
openshift-origin-broker	1.15.3.4-1
rubygem-openshift-origin-auth-remote-user	1.17.1-2
rubygem-openshift-origin-msg-broker-mcollective	1.17.8-1
rubygem-openshift-origin-admin-console	1.16.3.2-1
openshift-enterprise-yum-validator	2.0.1-1
openshift-enterprise-release	2.0.1-1
rubygem-openshift-origin-console	1.17.6.7-1
openshift-origin-console	1.15.1.5-1

Table 6.3: Software Versions–OpenShift Enterprise 2.0

Appendix F: Configuration Files

The following files are provided in the tarball with the reference architecture on the Red Hat customer portal.

```
# ls -Rth | grep -v total | grep -v 4.0K
.:
scripts
node1
ose-osp-ctrl
broker1

./scripts:
04-environment_config.sh
03-load_balancer.sh
02-net-config.sh
01-install-via-packstack-4-GA.sh
README
rhel-osp-4-GA-answer.txt
ifcfg-eth1

./node1:
openshift
mcollective

./node1/openshift:
iptables.filter.rules
iptables.nat.rules
node.conf
resource_limits.conf

./node1/mcollective:
server.cfg
facts.yaml

./ose-osp-ctrl:
openstack-dashboard
openshift-enterprise
cinder
ose_dib_env.sh
iptables
neutron
nova

./ose-osp-ctrl/openstack-dashboard:
local_settings

./ose-osp-ctrl/openshift-enterprise:
heat
dib

./ose-osp-ctrl/openshift-enterprise/heat:
neutron
```

```
./ose-osp-ctrl/openshift-enterprise/heat/neutron:
highly-available
OpenShift-1B1N-neutron.yaml

./ose-osp-ctrl/openshift-enterprise/heat/neutron/highly-available:
ose_ha_env.yaml
ose_node_env.yaml
ose_ha_stack.yaml
README.md
ose_ha_env.yaml.orig
ose_node_env.yaml.orig
ose_node_stack.yaml

./ose-osp-ctrl/openshift-enterprise/dib:
elements

./ose-osp-ctrl/openshift-enterprise/dib/elements:
openshift-enterprise-broker
openshift-enterprise-repos
openshift-enterprise-node

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-
broker:
install.d
element-deps
README.md

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-
broker/install.d:
30-openshift-enterprise-broker

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-repos:
pre-install.d

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-
repos/pre-install.d:
01-rhsm

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-node:
element-deps
install.d
README.md

./ose-osp-ctrl/openshift-enterprise/dib/elements/openshift-enterprise-
node/install.d:
30-openshift-enterprise-node

./ose-osp-ctrl/cinder:
cinder.conf

./ose-osp-ctrl/neutron:
fwaas_driver.ini
plugins
metadata_agent.ini
```

```
policy.json
rootwrap.conf
api-paste.ini
dhcp_agent.ini
lbaas_agent.ini
neutron.conf
release
l3_agent.ini
plugin.ini

./ose-osp-ctrl/neutron/plugins:
openvswitch

./ose-osp-ctrl/neutron/plugins/openvswitch:
ovs_neutron_plugin.ini

./ose-osp-ctrl/nova:
api-paste.ini
nova.conf
policy.json
release
rootwrap.conf

./broker1:
openshift
named
conf.d

./broker1/openshift:
plugins.d
broker.conf
console.conf

./broker1/openshift/plugins.d:
openshift-origin-admin-console.conf
openshift-origin-auth-remote-user.conf
openshift-origin-dns-nsupdate.conf
openshift-origin-msg-broker-mcollective.conf

./broker1/named:
dynamic
ose.example.com.key

./broker1/named/dynamic:
ose.example.com.db

./broker1/conf.d:
000002_openshift_origin_broker_proxy.conf
000002_openshift_origin_broker_servername.conf
ruby193-passenger.conf
```