Red Hat Reference Architecture Series

Deploying Sahara (Analytics-as-a-Service)

on Red Hat Enterprise Linux OpenStack Platform 5 [Technology Preview]

Jacob Liberman, Principal Software Engineer
RHCE

Version 1.0
October 2014

redhat

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

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

Send feedback to refarch-feedback@redhat.com

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/

NOTE: Sahara is a TECHNOLOGY PREVIEW in OpenStack Platform 5. More information on technology preview features can be found here: **https://access.redhat.com/solutions/21101**

Table of Contents

1 Executive Summary

Data analytics and data science are complicated endeavors requiring specialized tools and knowledge. Managing the tools necessary to perform analysis or experiments should be a trivial activity. However, data scientists lose valuable work cycles configuring their software ecosystem and waiting for time on a shared cluster. The Sahara project provides a simple means to provision Hadoop clusters in an OpenStack cloud infrastructure. It enables data processing on OpenStack and aims to eliminate tool management. Sahara saves data scientists valuable time in two ways. First, Sahara's cluster images reduce software management overhead because they are pre-configured with the required software and libraries. Second, OpenStack users can create virtual clusters to develop and test their applications as needed. Sahara improves time to solution by reducing scheduling conflicts on production clusters.

This reference architecture describes how to install and configure Sahara on Red Hat Enterprise Linux OpenStack Platform 5. It also shares deployment best practices for optimizing data processing tools such as Hadoop. The configuration and deployment steps described in this document focus on a single user to small group use case. Users have the ability to spin up a cluster from a predefined template which reads input from and writes output to Swift for long term storage. The cluster remains active for the duration of the activity. Additional clusters can be launched from the same template as needed.

Considerations for running a production cluster provisioned by Sahara are covered in a subsequent document.

2 Introduction

The Sahara project provides a scalable data processing stack and associated management technologies for OpenStack. Apache Hadoop is an industry standard MapReduce implementation. Sahara users can easily provision and manage Hadoop clusters on OpenStack. This allows Hadoop users to leverage the favorable economics of scale-out computing while at the same time reducing the complexity of cluster installation and management. Sahara is designed as an OpenStack component.

Sahara's key features include:

- Self-service cluster deployment
- A template-based framework for defining and managing clusters
- Support for various job types including MapReduce, Hive, Pig
- Support for various data sources including Swift and HDFS
- Integration with OpenStack services and management tools

This paper describes how to deploy Sahara on Red Hat Enterprise Linux OpenStack Platform (RHEL OSP) 5. It includes a description of Sahara's core components, a conceptual overview of how Sahara fits into the OpenStack software ecosystem, and complete steps for installing Sahara.

> **NOTE:** RHEL OSP 5 is based on the OpenStack Icehouse release. Sahara is available in Icehouse as a TECHNOLOGY PREVIEW. Sahara is expected to be fully integrated into RHEL OSP 6 based on the OpenStack Juno release. More information on technology preview features can be found here: **https://access.redhat.com/solutions/21101**

3 Architecture Overview

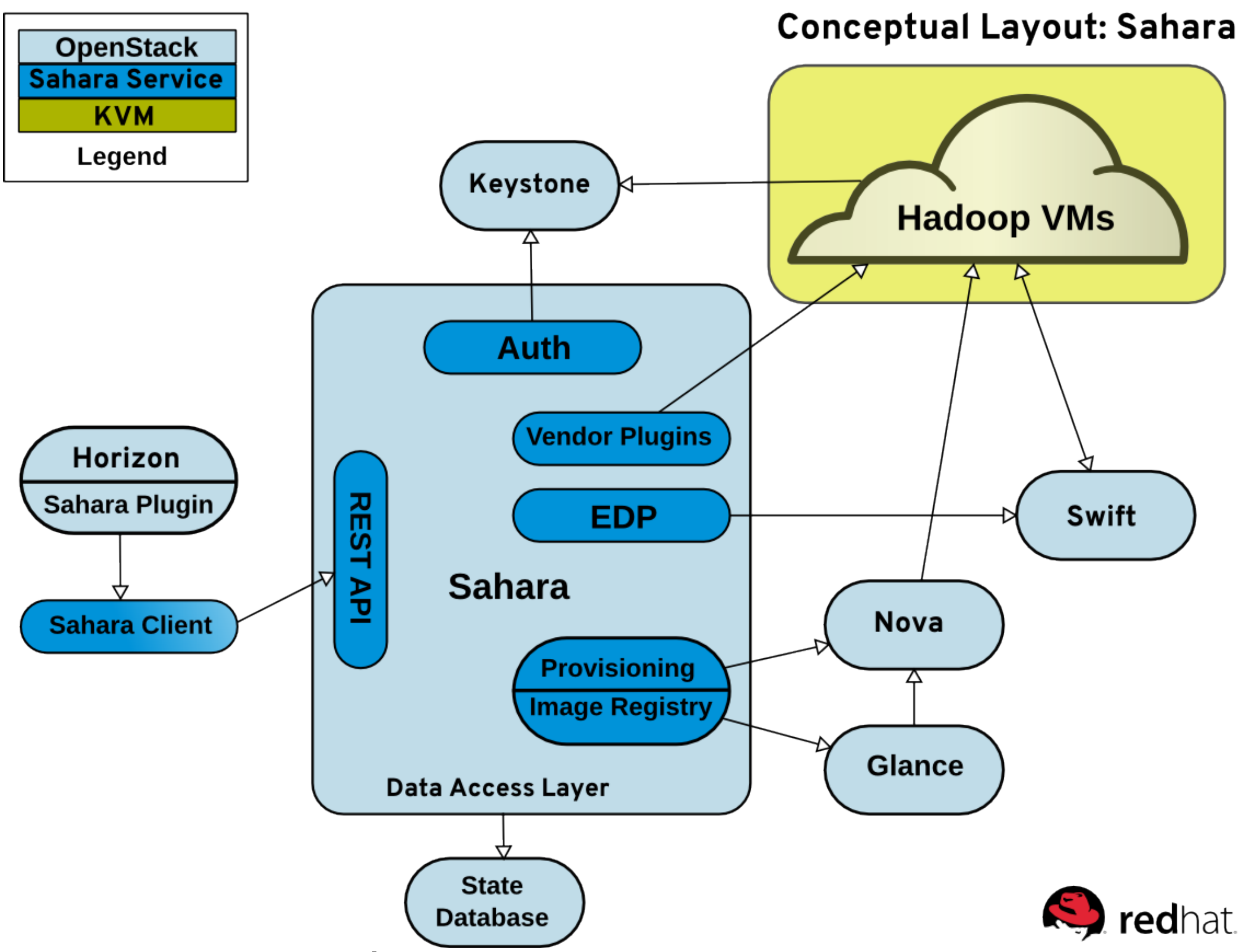
This section describes the reference architecture both physically and conceptually. It also defines terms as they are used in the rest of the document.

3.1 *OpenStack Sahara*

The Sahara project provides users with a simple means to define and provision Hadoop clusters within OpenStack. It is designed as an OpenStack component and managed through a REST API. Sahara supports several Hadoop distributions and versions and integrates with vendor-specific management tools.

3.2 *Solution Overview*

This section introduces Sahara and summarizes how it interacts with OpenStack. **Graphic 3.2.1: Sahara Conceptual Layout** depicts the relationships between services.



Graphic 3.2.1: Sahara Conceptual Layout

- **Auth** -- The Sahara auth component is responsible for client authentication and authorization. It communicates with Keystone as a service endpoint.
- **Data Access Layer** – The Data Access Layer stores persistent models in an internal database.
- **EDP** – The Elastic Data Processing (EDP) component schedules and manages Hadoop jobs on clusters provisioned by Sahara. The EDP component can use Swift as an input and/or output data store for Hadoop jobs.
- **Image Registry** – Image requirements for Sahara clusters depend on the Hadoop version and plugin. The Image Registry service interacts with Glance to ensure images meet the user's requirements when a cluster is formed.
- **Provisioning Engine** – The Sahara provisioning agent communicates with Nova, Neutron, Heat, Cinder and Glance to instantiate virtual machines and storage for Hadoop clusters.
- **REST API** – The Sahara REST API exposes Sahara functionality via REST to management tools such as the Python Sahara client.
- **Sahara Client** -- Sahara has a Python client similar to other OpenStack components.
- **Sahara Dashboard plugin** – The Sahara plugin allows users to manage Sahara via the web-based OpenStack Dashboard.
- **Vendor Plugins** – These are pluggable mechanisms responsible for configuring and launching Hadoop on provisioned virtual machines.

Refer to **OpenStack Sahara documentation** for more information regarding Sahara architecture.

3.3 Software Component Overview

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

3.3.1 Red Hat Enterprise Linux 7

Red Hat Enterprise Linux (RHEL) 7 is the base operating system used in this reference architecture. All servers run RHEL 7, the latest major release of Red Hat's flagship platform. RHEL 7 provides a stable code base for bare metal servers, virtual machines, Infrastructure-as-a-Service (IaaS), and Platform-as-a-Service (PaaS) across the enterprise data center.

3.3.2 Apache Hadoop

The Apache Hadoop software library is a framework allowing the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to be scalable to thousands of machines. The Hadoop core modules and related projects mentioned in this reference architecture are described below.

3.3.2.1 Hadoop Common

The common utilities that support other Hadoop modules.

3.3.2.2 Hadoop Distributed File System (HDFS)

A fault-tolerant and scalable distributed file system. HDFS provides streaming access for large data sets. Data replication schemes and error detection ensure quick recovery from hardware failure.

3.3.2.3 Hadoop YARN

YARN is a resource manager that acts as a data operating system for Hadoop. YARN allows multiple applications to run natively in Hadoop.

3.3.2.4 Hadoop MapReduce

Hadoop MapReduce is a software framework for writing applications that process large amounts of data in parallel on clusters of commodity hardware. MapReduce jobs split input data into chunks that can be operated on independently by map tasks. The output of these jobs are input to reduce tasks that recombine the data to yield a result. MapReduce is a dominant paradigm for large-scale data analysis.

3.3.2.5 Apache Oozie

Oozie is a scalable and extensible workflow scheduler system for Hadoop jobs. It is integrated with Hadoop to support several types of jobs including Java, MapReduce and Streaming MapReduce, Pig, and Hive.

3.3.2.6 Apache Pig

Pig consists of a high level language for expressing and evaluating large scale data sets. Pig simplifies the tasks of MapReduce parallel programming.

3.3.3 RHEL OpenStack Platform 5

OpenStack is open source software for building private and public clouds. Red Hat is an active contributor to the OpenStack code base[1]. RHEL OSP 5 combines the benefits of Red Hat's OpenStack technology, Kernel-based Virtual Machine (KVM), and Red Hat Enterprise Linux. RHEL OSP 5 is based on the ninth OpenStack release code named "Icehouse". RHEL OSP 5 is certified to run on both Red Hat Enterprise Linux 6.5 and 7.

> **NOTE:** See the product release notes for more information:
> **https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/5/html/Release_Notes/**

1 http://www.redhat.com/infographics/openstack/

Sahara integrates with the following OpenStack services:

3.3.4 Identity Service ("Keystone")

This is the central authentication and authorization mechanism for all OpenStack users and services. Sahara is accessible to other OpenStack services via a Keystone service endpoint. Furthermore, Keystone tenants manage access to Sahara resources including clusters, nodes, and templates.

3.3.5 Image Service ("Glance")

This service registers and delivers virtual machine images. Sahara cluster nodes can be installed via pre-built images of vanilla Apache Hadoop or vendor supplied images. Users can interact with the pre-built images via the Sahara API once they are registered with Glance.

3.3.6 Compute Service ("Nova")

The Compute service provisions and manages large networks of virtual machines. Sahara leverages the Nova framework to define and build Hadoop clusters via user defined templates. The templates include node and cluster specific information such as the plugin name, Hadoop version, and Nova hardware flavor. Sahara node group templates are passed to Nova along with a Glance image name to instantiate virtual machine nodes. Cluster templates combine the nodes into clusters.

3.3.7 Network Service ("Neutron")

The Network service is a scalable API-driven service for managing networks and IP addresses. Neutron gives users self-service control over their network configurations. Sahara plugins define the network access method for cluster nodes. They are accessed most commonly via Neutron floating IP addresses assigned to the instances during creation.

3.3.8 Object Storage Service ("Swift")

The Object Storage service provides a fully distributed, API-accessible storage platform that can be integrated into applications or used for backup, archiving, and data retention. Swift data stores can be used within Hadoop data processing jobs after the application of a patch. Sahara automatically sets information about the Swift filesystem implementation, location awareness, and tenant name for authorization. The only information required when launching a Hadoop job with a Swift backend are the username and password to access Swift.

3.3.9 Dashboard Service ("Horizon")

The Dashboard service is an extensible web-based application that allows cloud administrators and users to control and provision compute, storage, and networking resources. A Sahara plugin allows administrators to deploy and manage Sahara clusters via the Dashboard.

3.4 Server Roles

This section defines the various server roles used in this reference architecture.

3.4.1 Cloud Controller

Cloud controller is the designation used in this reference architecture for the server that provides the endpoint for REST-based API queries to the majority of the OpenStack services. These include Compute, Image, Identity, Block, Network, and Data processing. Although RHEL OSP allows for multiple, high availability cloud controllers, only one cloud controller is used in this reference architecture.

3.4.2 Compute Node

Compute node refers to an OpenStack server that runs a KVM hypervisor. It is responsible for running virtual machine instances. In this reference architecture, Hadoop clusters are instantiated across multiple compute nodes. By default a new instance is spawned on the compute node with the most free memory in a round robin fashion.

3.4.3 Network Node

The Network Node provides centralized networking control for all compute nodes and tenants. It runs the various Neutron agents that control L3 networking functionality in the cluster. In this reference architecture, the network agents run on a single dedicated server. It is also possible to run these agents on the cloud controller or multiple network nodes in a clustered fashion.

3.4.4 Swift Servers

Understanding the Swift ring is central to understanding the role of a server in the Swift cluster. A Swift ring represents a mapping between the names of entities stored on disk and their physical location. There are separate rings for accounts, containers, and objects. When a component needs to interact with an object, container, or account, it interacts with the appropriate ring to determine the target's location in the cluster.

3.4.4.1 Swift Proxy Server

This server ties together the Swift architecture. For each request, the proxy server looks up the location of the account, object, or container in the appropriate Swift ring and routes the request accordingly. In this reference architecture, the cloud controller also acts as the Swift proxy server.

3.4.4.2 Swift Storage Server

The Swift storage servers are simple blob storage servers that can store, retrieve, and delete objects stored on local devices. Objects are stored using a path derived from the object's name and a time stamp. In this reference architecture there are three dedicated Swift storage servers. They are configured as three-way replication partners to ensure data coherency.

3.5 Network Names

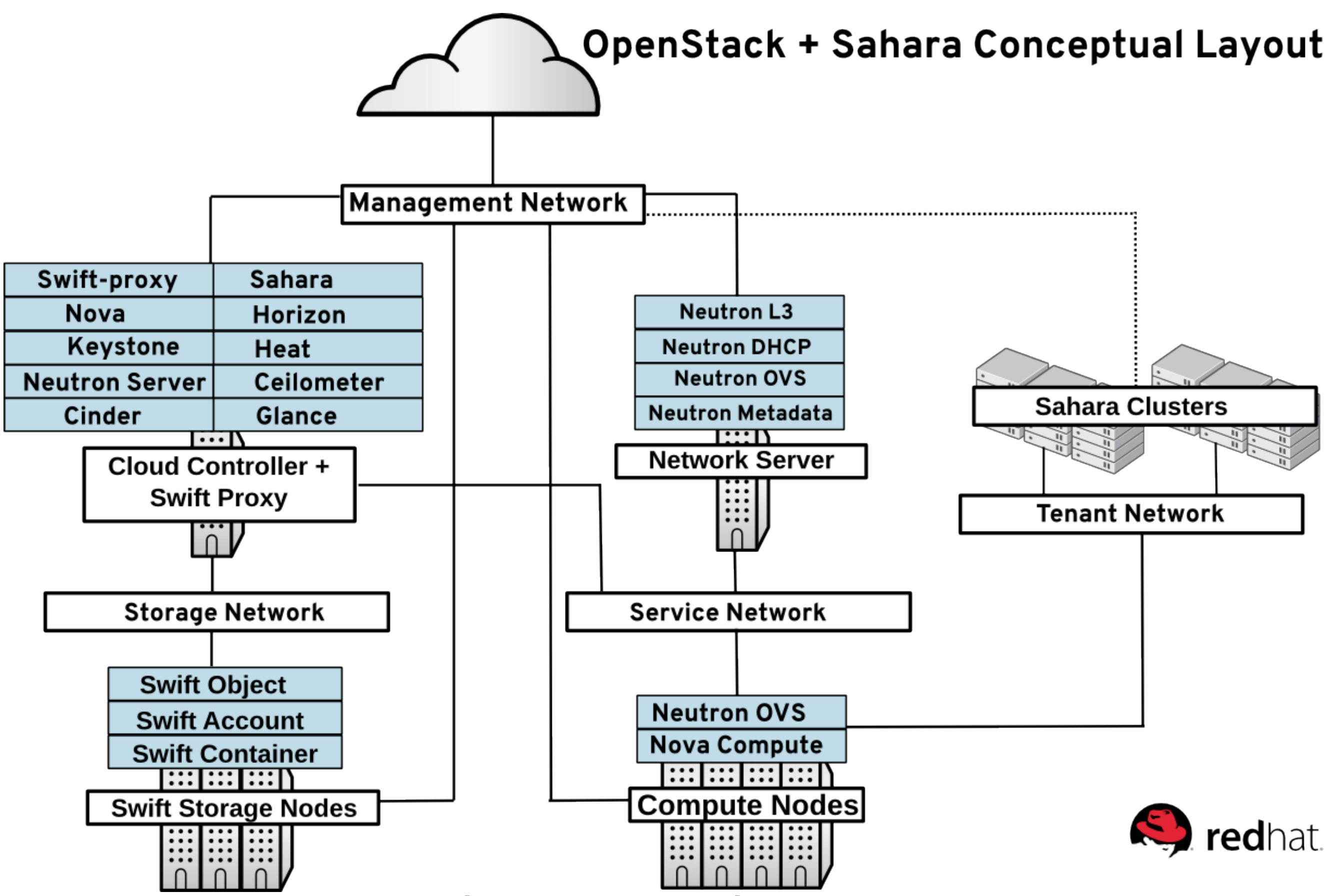
A typical OpenStack deployment includes several network roles. In some cases the roles overlap across the same physical interfaces or switches. In others each role has a dedicated network interface and switches. This reference architecture uses the following networks:

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

1. **Service network** – this network exposes the OpenStack APIs. It also handles inter-service communication between the OpenStack services and schedulers.
2. **Tenant network** – virtual machines communicate over this network within the cloud deployment. The addressing requirements of this network depend on the plugin that is used.
3. **Storage Network** – this network is dedicated for storage traffic between the Swift servers and the OpenStack servers.

3.6 Conceptual Diagram of the Solution Stack

Graphic 3.6.1: Service Placement depicts the solution stack including networks, server roles, and service placement. Section **4 Configuration Details** shares complete details.



Graphic 3.6.1: Service Placement

3.7 Sahara Terminology

Sahara uses collections of simple objects to define and execute Hadoop jobs. The objects are stored in the Sahara database and can be reused on subsequent jobs. A job consists of:

1. At least one executable binary
2. Optional supporting libraries
3. input data
4. output location
5. additional configuration values needed to run the job

The job binary object stores a resource locator to a single JAR file or script and the credentials needed to retrieve it. A job refers to the binary when it is launched and all its supporting libraries. The job and job binary are analogous to an instance and an image: multiple jobs can share the same job binary just as multiple instances can be spawned from the same image. The data source objects designate the location of any input or output data required or generated by the job. Sahara supports both Swift and HDFS as data sources.

3.8 Submitting Sahara Jobs

This section describes the typical workflow for running Sahara jobs. It also describes the typical workflow for a single job.

> **NOTE:** More information on these topics can be found in the Sahara documentation: **http://docs.openstack.org/developer/sahara/userdoc/edp.html**.

3.8.1 Sahara Workflow

A lifecycle of a Sahara job follows a general workflow:

- Define cluster node templates that reference a Hadoop version, an image, and the number and type of Hadoop services to run
- Define a cluster template that references cluster node templates
- Launch a cluster from the cluster template
- Once the cluster is instantiated, create all the job binaries needed to run the job.
- Store the job binaries in the Sahara database or Swift
- Create a job that references the job binaries
- Create an input data source that references the data to be processed
- Create an output data source
- Launch the job to the cluster.

Existing objects simplify the workflow. New jobs can reference existing job binaries and already instantiated clusters.

3.8.2 Submitting Sahara Jobs

Hadoop jobs can be submitted to Sahara in several ways:

1. The user can login to the cluster and submit jobs interactively through command line interfaces for Hadoop and Oozie
2. The user can submit a job to the Oozie server via an Oozie client or the Oozie REST API
3. The user can submit a job through the Sahara client or web UI.

The third method highlights several features of using Sahara. First, job binaries and data sources can be combined into reusable templates. Users can re-run the same job multiple times via the template or share it with different users.

Next, jobs submitted through the Sahara EDP workflow can be run on transient clusters. Transient clusters are created specifically for the job and shut down once the job is finished.

Section **6 Validate and Test Sahara** demonstrates the first and third job submission methods.

4 Configuration Details

This section of the paper describes the hardware and software used to configure the reference architecture in the Red Hat Solutions Engineering lab. It includes security details.

4.1 Environment

The reference architecture environment consists of the components required to build a small Red Hat Enterprise Linux OpenStack Platform cloud infrastructure. It includes small form factor servers for the OpenStack servers and Swift storage servers with more internal storage capacity.

4.1.1 Service Placement

Table 4.1.1.1: Service Placement lists the service placement for all OpenStack services. The cloud controller runs the majority of services.

Host	Role	Service
rhos0	Cloud controller	rabbitmq
		neutron-server
		openstack-ceilometer-alarm-evaluator
		openstack-ceilometer-alarm-notifier
		openstack-ceilometer-api
		openstack-ceilometer-central
		openstack-ceilometer-collector
		openstack-ceilometer-notification
		openstack-cinder-api
		openstack-cinder-scheduler
		openstack-cinder-volume
		openstack-glance-api
		openstack-glance-registry
		openstack-heat-api-cfn
		openstack-heat-api
		openstack-heat-engine
		openstack-keystone
		openstack-nova-api
		openstack-nova-cert

		openstack-nova-conductor
		openstack-nova-consoleauth
		openstack-nova-novncproxy
		openstack-nova-scheduler
		openstack-sahara-api
		openstack-swift-proxy
rhos1	Network server	neutron-dhcp-agent
		neutron-l3-agent
		neutron-metadata-agent
		neutron-openvswitch-agent
		neutron-ovs-cleanup
rhos[2-5]	Compute nodes	neutron-openvswitch-agent
		neutron-ovs-cleanup
		openstack-nova-compute
rhos[7-9]	Storage nodes	swift-account-reaper
		swift-account-auditor
		swift-container-replicator
		swift-container-auditor
		swift-object-auditor
		swift-object-updater
		swift-account-replicator
		swift-object-replicator
		swift-container-updater

Table 4.1.1.1: Service Placement

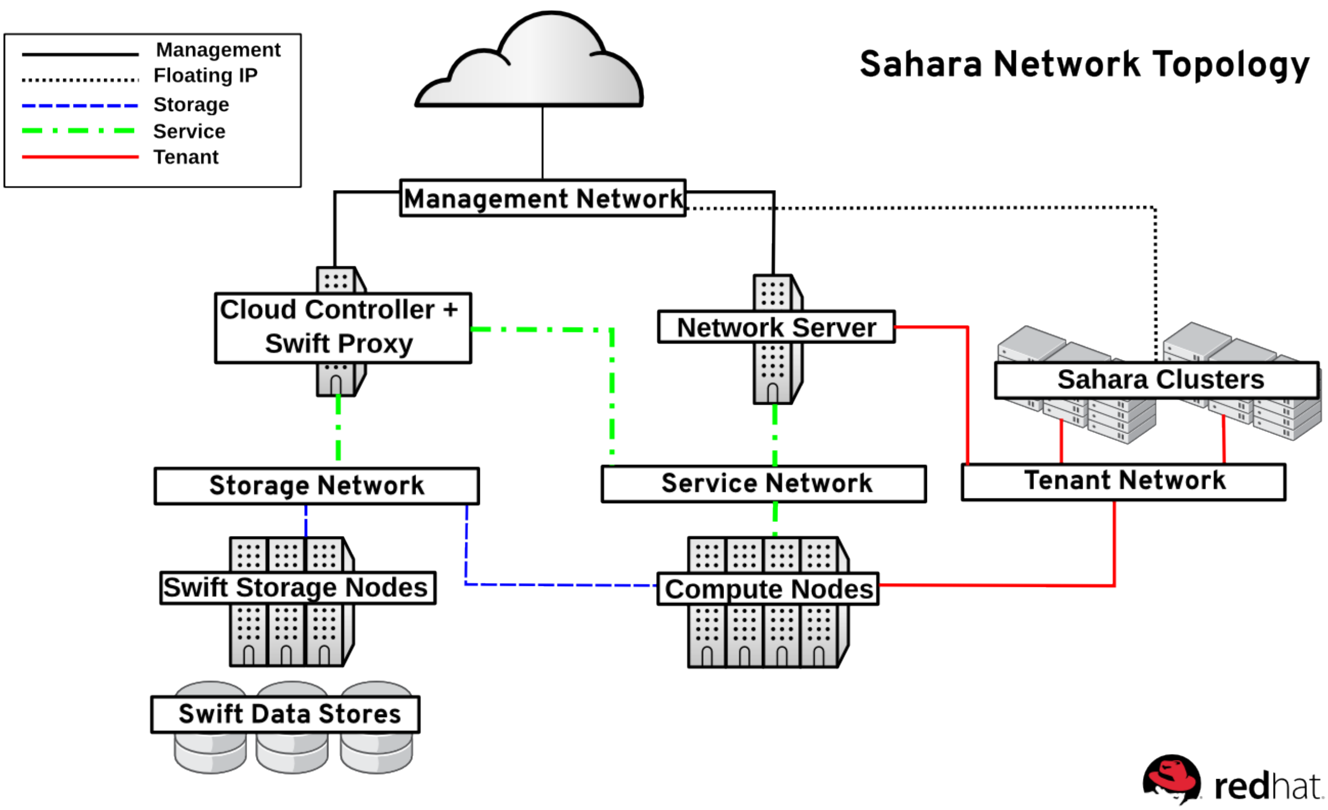
4.1.2 Network Topology

Graphic 4.1.2.1: Network Topology shows the network topology of this reference architecture.

- All nine servers communicate via the lab network switch on the management network. The management network uses v4 IP addresses in the 10.19.137.0/24 range.
- The tenant network carries communication between virtual machines and software-defined networking components. It is the private network over which the instances communicate. In this reference architecture, a network switch connected to 10 GB interfaces on the compute nodes is tagged to VLAN IDs 1000:1010 for tenant communication. They do not have IP addresses.

NOTE: The tenant network carries tenant network traffic over tagged VLANs. The interfaces connected to this network are not configured with IPv4 addresses by the OpenStack administrator. Instead, instances and services are allocated addresses within user-created subnets on tenant networks. Network namespaces prevent different users' subnets from conflicting with each other or with the infrastructure's own subnets.

- All Swift storage communication occurs via a second 10Gb storage network switch on the 172.31.0.0/16 network. This network delivers the Object storage service communication and delivery.
- The Service network carries service requests to the service listeners. These include the various schedulers and agents deployed in the OpenStack environment. The service traffic is segmented from the tenant and management traffic. The service network interfaces are assigned v4 IP addresses in the 172.16.2.0/24 range.



Graphic 4.1.2.1: Network Topology

NOTE: This reference architecture uses four physical networks. However it is possible to deploy supported OpenStack solutions with more or fewer networks.

4.1.3 Network Addresses

Table 4.1.3.1: Host IP Addresses lists the IPv4 Addresses used in this reference architecture by server host name and role. All servers have interfaces on four networks: management, service, tenant, and storage. There are four roles: cloud controller, Neutron networker, Swift storage server, or compute node. A minimum of one cloud controller, one Neutron networker, three Swift storage servers, and one compute node are required to implement this reference architecture. In this example four compute nodes were used.

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

Host	Role	Network	Interface	Network Address
rhos0	Cloud Controller	Management	eno1	10.19.137.100
		Storage	enp20	172.31.139.100
		Service	eno2	172.16.2.100
rhos1	Neutron Networker	Management	eno1	10.19.137.101
		Storage	enp20	172.31.139.101
		Tenant	enp21	VLAN 1000:1010
		Service	eno2	172.16.2.101
rhos2	Compute Node	Management	eno1	10.19.137.102
		Storage	enp20	172.31.139.102
		Tenant	enp21	VLAN 1000:1010
		Service	eno2	172.16.2.102
rhos3	Compute Node	Management	eno1	10.19.137.103
		Storage	enp20	172.31.139.103
		Tenant	enp21	VLAN 1000:1010
		Service	eno2	172.16.2.103
rhos4	Compute Node	Management	eno1	10.19.137.104
		Storage	enp20	172.31.139.104
		Tenant	enp21	VLAN 1000:1010
		Service	eno2	172.16.2.104
rhos5	Compute Node	Management	eno1	10.19.137.105
		Storage	enp20	172.31.139.105

		Tenant	enp21	VLAN 1000:1010
		Service	eno2	172.16.2.105
rhos7	Compute Node	Management	em1	10.19.137.107
		Storage	p1p2	172.31.139.107
rhos8	Compute Node	Management	em1	10.19.137.108
		Storage	p1p2	172.31.139.108
rhos9	Compute Node	Management	em1	10.19.137.109
		Storage	p1p2	172.31.139.109

Table 4.1.3.1: Host IP Addresses

4.2 Software Configuration

This section of the reference architecture lists the required software revisions. It also lists software configuration details related to security. Customers who use the correct OpenStack and Red Hat Enterprise Linux channels on Red Hat Network (RHN) or Subscription Manager meet the minimum required software versions.

4.2.1 Required Channels

Red Hat Enterprise Linux OpenStack Platform is available via the Red Hat Network channels and RHN Certificate Server repositories listed in **Table 4.2.1.1: Required Channels**.

Channel	Source
rhel-x86_64-server-6	RHN Classic
rhel-x86_64-server-6-ost-3	RHN Classic
rhel-6-server-rpms	RHN Certificate
rhel-6-server-openstack-5.0-rpms	RHN Certificate
rhel-6-server-rh-common-rpms	RHN Certificate

Table 4.2.1.1: Required Channels

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

4.2.2 Software Versions

Table 4.2.2.1: Software Versions lists the software versions used to develop this reference architecture.

Host	Software	Version
	openstack-ceilometer-alarm	2014.1.2-1

Cloud Controller	openstack-ceilometer-api	2014.1.2-1
	openstack-ceilometer-central	2014.1.2-1
	openstack-ceilometer-collector	2014.1.2-1
	openstack-ceilometer-common	2014.1.2-1
	openstack-ceilometer-notification	2014.1.2-1
	openstack-cinder	2014.1.2-2
	openstack-dashboard	2014.1.2-1
	openstack-dashboard-theme	2014.1.2-1
	openstack-glance	2014.1.2-1
	openstack-heat-api	2014.1.2-1.0
	openstack-heat-api-cfn	2014.1.2-1.0
	openstack-heat-api-cloudwatch	2014.1.2-1.0
	openstack-heat-common	2014.1.2-1.0
	openstack-heat-engine	2014.1.2-1.0
	openstack-keystone	2014.1.2.1-1
	openstack-neutron	2014.1.2-2
	openstack-neutron-ml2	2014.1.2-2
	openstack-nova-api	2014.1.2-1
	openstack-nova-cert	2014.1.2-1
	openstack-nova-common	2014.1.2-1
	openstack-nova-conductor	2014.1.2-1
	openstack-nova-console	2014.1.2-1
	openstack-nova-novncproxy	2014.1.2-1
	openstack-nova-scheduler	2014.1.2-1
	openstack-sahara	2014.1.2-1
	openstack-selinux	0.5.14-4
	openstack-swift	1.13.1-3
	openstack-swift-plugin-swift	1.7-3
	openstack-swift-proxy	1.13.1-3
	openstack-utils	2014.1-3.2
	python-django-openstack-auth	1.1.5-2
	python-django-sahara	2014.1.2-1
	python-neutron	2014.1.2-2

	python-neutronclient	2.3.4-2
	python-saharaclient	0.7.0-3
	python-swiftclient	2.1.0-2
	redhat-access-plugin-openstack	5.0.0-3
Compute Node	openstack-neutron	2014.1.2-2
	openstack-neutron-openvswitch	2014.1.2-2
	openstack-nova-common	2014.1.2-1
	openstack-nova-compute	2014.1.2-1
	openstack-selinux	0.5.14-4
	openstack-utils	2014.1-3.2
	python-neutron	2014.1-3.2
	python-neutronclient	2.3.4-2
Neutron Networker	openstack-neutron	2014.1.2-2
	openstack-neutron-openvswitch	2014.1.2-2
	openstack-selinux	0.5.14-4
	openstack-utils	2014.1-3.2
	python-neutron	2014.1-3.2
	python-neutronclient	2.3.4-2
Swift Storage Server	openstack-selinux	0.5.14-4
	openstack-swift	0.5.14-4
	openstack-swift-account	1.13.1-3
	openstack-swift-container	1.13.1-3
	openstack-swift-object	1.13.1-3
	python-swiftclient	2.1.0-2

Table 4.2.2.1: Software Versions

4.3 Security Details

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

4.3.1 Firewall Configuration

Table 4.3.1.1: Allowed Ports by Role lists the allowed ports by host, and role. Implement these via either **`iptables`** or **`firewalld`**.

Port	Host	Role

22, 80, 443, 873, 3260, 3306, 5000, 5671-5672, 6080, 8000, 8003-8004, 8080, 8386, 8773-8777, 9191, 9292, 9696, 9697, 11211, 35357	rhos0	Cloud Controller
22, 4789	rhos1	Neutron Networker
22, 53, 67, 4789, 5900:5999	rhos[2-5]	Compute node
22, 873, 6000-6002	rhos[7-9]	Storage node

Table 4.3.1.1: Allowed Ports by Role

4.3.2 SELinux Configuration

Red Hat Enterprise Linux OpenStack Platform supports **`SELinux`** in **enforcing** mode in Red Hat Enterprise Linux 7. **Table 4.3.2.1: Supported SELinux Package Versions** lists the required packages.

Package	Version
libselinux	2.2.2-6
selinux-policy	3.12.1-153
selinux-policy-targeted	3.12.1-153
openstack-selinux	0.5.14-4

Table 4.3.2.1: Supported SELinux Package Versions

4.4 Hardware Details

4.4.1 Server Hardware Configuration

Table 4.4.1.1: Server Hardware lists the hardware specifications for the servers used in this reference architecture.

> **NOTE:** Red Hat Enterprise Linux OpenStack Platform servers do not require identical hardware. The hardware used in this reference architecture meets the minimum requirements outlined in the OpenStack documentation. The **Red Hat Enterprise Linux OpenStack Platform 5** installation guide contains further details.

Hardware	Specifications
6 x IBM BladeCenter HS22	2 x Intel(R) Xeon(R) CPU X5680 @ 3.33 GHz (6 core)
	2 x Broadcom NetXtreme II BCM5709S Gb Ethernet 2 x Emulex Corporation OneConnect 10Gb NIC
	6 x DDR3 8192 MB @1333 MHZ DIMMs

	2 x 146GB SAS internal disk drives
3 x Dell PowerEdge R520	2 x Intel(R) Xeon(R) CPU X5650 @ 2.67 GHz (6 core)
	2 x Broadcom NetXtreme II BCM5709S Gb Ethernet 2 x Emulex Corporation OneConnect 10Gb NIC
	6 x DDR3 8192 MB @1333 MHZ DIMMs
	12 x 146GB SAS internal disk drives

Table 4.4.1.1: Server Hardware

4.4.2 Network Hardware

Each server has up to four interfaces: a management network interface, a private interface for service communication, an interface with tagged VLANs for tenant traffic, and a storage interface.

The management network interfaces and network storage interfaces are connected via an HP Procurve 5400 switch. The network storage interfaces are uplinked through a 10GB switch module to 10 GB ports on the Procurve switch.

5 Implementing Sahara on OpenStack

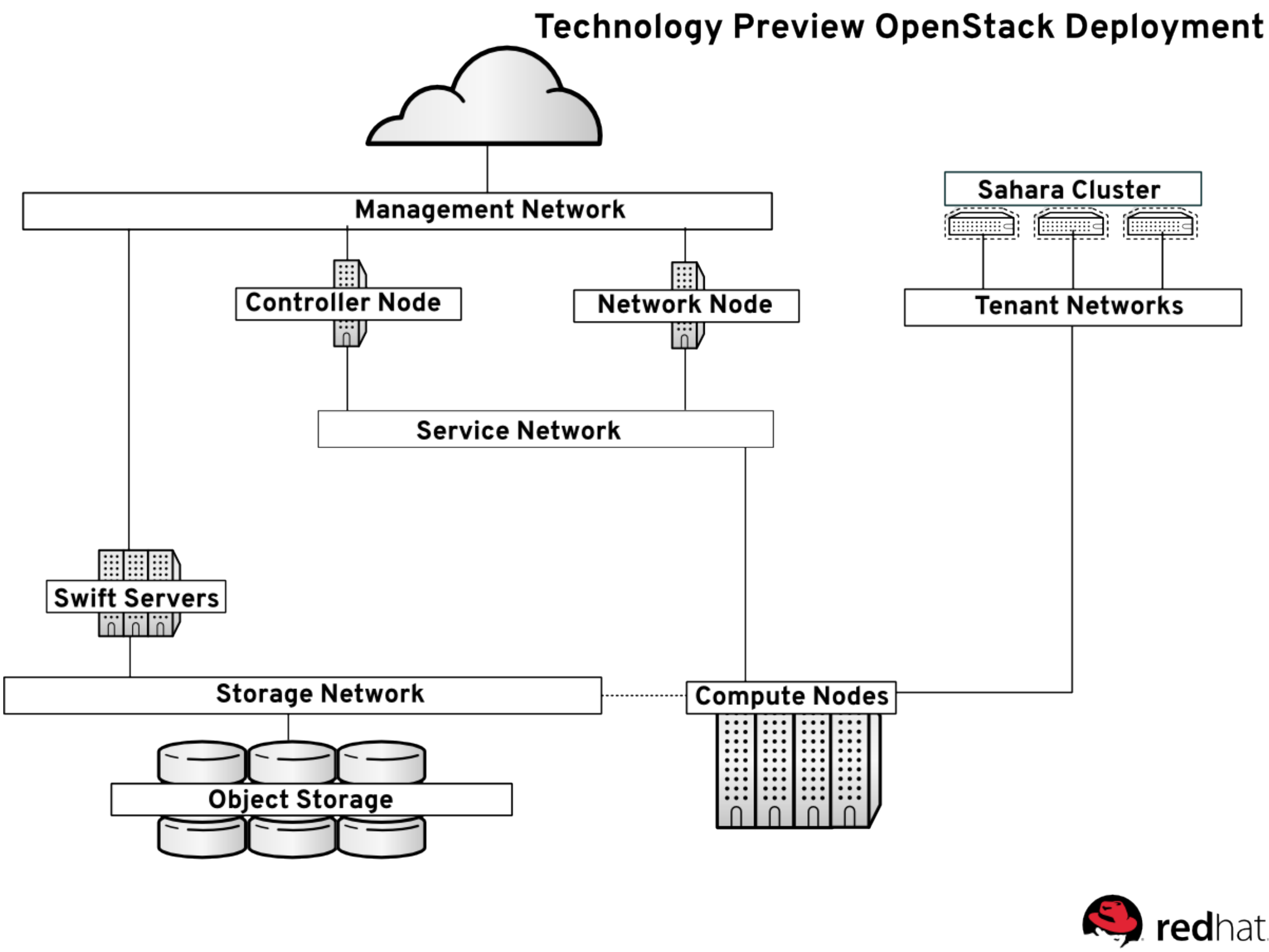
This section describes the process that was followed to stand up Sahara on OpenStack in the Systems Engineering lab.

5.1 Prepare the Hosts

Complete the steps described in this section on all servers including the cloud controller, network server, compute nodes, and storage servers.

5.2 Deploy OpenStack

RHEL OSP 5 supports a number of Puppet-based OpenStack installers. This reference architecture demonstrates Sahara on a small but realistic non-HA OpenStack deployment depicted in **Graphic 5.2.1: OpenStack Deployment**.



Graphic 5.2.1: OpenStack Deployment

The reference architecture OpenStack deployment can be summarized as follows:

- All servers run Red Hat Enterprise Linux 7 deployed via a Red Hat Network Satellite server.
- All servers run Red Hat Enterprise Linux OpenStack Platform 5, based on the “Icehouse” release configured via Puppet manifests. The Puppet parameters are listed by server role in **Appendix C: Host Group YAML Output**.
- Core OpenStack services configured in this reference architecture include: Keystone, Nova, Horizon, Ceilometer, Heat, Cinder, and Glance.
- Neutron networking is used in this reference architecture with ML2 and Open vSwitch plugins.
- VLAN 1000:1010 are used for tenant networking.
- Swift Object Storage is used for both Glance images and as a Sahara data store. The cloud controller acts as the Swift proxy.
- Each server has four network interfaces and dedicated networks for management, storage, service, and tenant traffic.

NOTE: Appendix A: References links to installation documentation for Red Hat Enterprise Linux 7 and Red Hat Enterprise Linux OpenStack Platform 5.

5.3 Prepare the Environment

1. On the cloud controller, source the *keystonerc_admin* file.

```
[root@rhos0 ~]# source /root/keystonerc_admin

[root@rhos0 ~(openstack_admin)]# env | grep OS_
OS_PASSWORD=redhat
OS_AUTH_URL=http://172.16.2.100:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

2. Create an external network in the ***services*** tenant named ***ext_net*** that maps to the physical lab network.

```
[root@rhos0 ~(openstack_admin)]# neutron net-create ext_net   \
--router:external=True --provider:network_type flat \
--provider:physical_network physext

Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
```

```
| id                        | 97680250-36e7-49be-bdcd-44c6ac86b0d1 |
| name                      | ext_net                              |
| provider:network_type     | flat                                 |
| provider:physical_network | physext                              |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 0f07e934bfcc4aa185f854222bfc0c5a     |
+---------------------------+--------------------------------------+
```

3. Create a subnet named ***public*** with floating IP addresses and a default gateway.

```
[root@rhos0 ~(openstack_admin)]# neutron subnet-create --name public  \
--gateway 10.19.143.254 --allocation-pool \
start=10.19.137.111,end=10.19.137.119 ext_net 10.19.136.0/21 -- \
--enable_dhcp=False --dns_nameservers list=true 10.19.143.247

Created a new subnet:
+------------------+----------------------------------------------------+
| Field            | Value                                              |
+------------------+----------------------------------------------------+
| allocation_pools | {"start": "10.19.137.111", "end": "10.19.137.119"} |
| cidr             | 10.19.136.0/21                                     |
| dns_nameservers  | 10.19.143.247                                      |
| enable_dhcp      | False                                              |
| gateway_ip       | 10.19.143.254                                      |
| host_routes      |                                                    |
| id               | 64826389-70e9-4f78-b42b-af673ddf536f               |
| ip_version       | 4                                                  |
| name             | public                                             |
| network_id       | 97680250-36e7-49be-bdcd-44c6ac86b0d1               |
| tenant_id        | 0f07e934bfcc4aa185f854222bfc0c5a                   |
+------------------+----------------------------------------------------+
```

4. Create a tenant and tenant user.

```
[root@rhos0 ~(openstack_admin)]# keystone user-create --name refarch \
--pass refarch
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 418410f6b8cd451481fb86904cb42758 |
|   name   |             refarch              |
| username |             refarch              |
+----------+----------------------------------+

[root@rhos0 ~(openstack_admin)]# keystone tenant-create \
--name refarch-tenant
+-------------+----------------------------------+
|   Property  |              Value               |
```

```
+-------------+-----------------------------------+
| description |                                   |
|   enabled   |                True               |
|      id     | 7f367eecfda24aabb15f401df8cd1b70 |
|     name    |           refarch-tenant          |
+-------------+-----------------------------------+
```

5. Add the ***_member_*** role to the tenant user.

```
[root@rhos0 ~(openstack_admin)]# keystone user-role-add --user-id refarch
--tenant-id refarch-tenant --role-id _member_

[root@rhos0 ~(openstack_admin)]# keystone user-role-list --user-id refarch
--tenant-id refarch-tenant
+----------------------------------+----------------------------------------+
|                id                |   name    |              user_id
|            tenant_id             |
+----------------------------------+----------------------------------------+
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
418410f6b8cd451481fb86904cb42758 | 7f367eecfda24aabb15f401df8cd1b70 |
+----------------------------------+----------------------------------------+
```

6. Create a keystonerc file for the tenant user.

```
[root@rhos0 ~(openstack_admin)]# cat /root/keystonerc_refarch << EOF
export OS_USERNAME=refarch
export OS_TENANT_NAME=refarch-tenant
export OS_PASSWORD=refarch
export OS_AUTH_URL=http://10.19.137.100:35357/v2.0/
export PS1='[\u@\h \W(refarch_member)]\$ '
EOF
```

7. Switch to ***refarch*** user.

```
[root@rhos0 ~(openstack_admin)]# source /root/keystonerc_refarch

[root@rhos0 ~(refarch_member)]# env | grep OS_
OS_PASSWORD=refarch
OS_AUTH_URL=http://10.19.137.100:35357/v2.0/
OS_USERNAME=refarch
OS_TENANT_NAME=refarch-tenant
```

8. Create a tenant network named ***net1*** and a subnet named ***private***.

```
[root@rhos0 ~(refarch_member)]# neutron net-create net1

Created a new network:
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| admin_state_up | True                                 |
| id             | 0c0bc665-c992-4594-b7c5-7b94690e94e3 |
| name           | net1                                 |
```

```
| shared           | False                                 |
| status           | ACTIVE                                |
| subnets          |                                       |
| tenant_id        | 7f367eecfda24aabb15f401df8cd1b70      |
+-----------------+--------------------------------------+

[root@rhos0 ~(refarch_member)]# neutron subnet-create --name private net1 \
172.16.3.0/24 --dns_nameservers list=true 10.19.143.247
Created a new subnet:
+------------------+------------------------------------------------+
| Field            | Value                                          |
+------------------+------------------------------------------------+
| allocation_pools | {"start": "172.16.3.2", "end": "172.16.3.254"} |
| cidr             | 172.16.3.0/24                                  |
| dns_nameservers  | 10.19.143.247                                  |
| enable_dhcp      | True                                           |
| gateway_ip       | 172.16.3.1                                     |
| host_routes      |                                                |
| id               | fdba629e-7ece-4a8f-af18-43fe1e57c957           |
| ip_version       | 4                                              |
| name             | private                                        |
| network_id       | 0c0bc665-c992-4594-b7c5-7b94690e94e3           |
| tenant_id        | 7f367eecfda24aabb15f401df8cd1b70               |
+------------------+------------------------------------------------+
```

9. View the network and subnet.

```
[root@rhos0 ~(refarch_member)]# neutron net-list
+--------------------------------------+---------+--------------+---------+
| id                                   | name    | subnets
+--------------------------------------+---------+--------------+---------+
| 0c0bc665-c992-4594-b7c5-7b94690e94e3 | net1    | fdba629e-7ece-4a8f-af18-
43fe1e57c957 172.16.3.0/24 |
| 97680250-36e7-49be-bdcd-44c6ac86b0d1 | ext_net | 64826389-70e9-4f78-b42b-
af673ddf536f                |
+--------------------------------------+---------+--------------+---------+

[root@rhos0 ~(refarch_member)]# neutron subnet-list
+--------------------------------------+---------+--------------+---------+
| id                                   | name    | cidr          |
allocation_pools                                |
+--------------------------------------+---------+--------------+---------+
| fdba629e-7ece-4a8f-af18-43fe1e57c957 | private | 172.16.3.0/24 | {"start":
"172.16.3.2", "end": "172.16.3.254"} |
+--------------------------------------+---------+--------------+---------+
```

10. Create a router named ***route1***.

```
[root@rhos0 ~(refarch_member)]# neutron router-create route1
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
```

```
| external_gateway_info |                                      |
| id                    | 643ebbfe-2186-4272-ae10-46444e5d128b |
| name                  | route1                               |
| status                | ACTIVE                               |
| tenant_id             | 7f367eecfda24aabb15f401df8cd1b70     |
+-----------------------+--------------------------------------+

[root@rhos0 ~(refarch_member)]# neutron router-show route1
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 643ebbfe-2186-4272-ae10-46444e5d128b |
| name                  | route1                               |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | 7f367eecfda24aabb15f401df8cd1b70     |
+-----------------------+--------------------------------------+
```

11. Attach the ***private*** subnet to the router.

```
[root@rhos0 ~(refarch_member)]# subnet_id=$(neutron subnet-list | \
awk ' /172.16.3.0/ { print $2 } ')

[root@rhos0 ~(refarch_member)]# echo -e "$subnet_id"
fdba629e-7ece-4a8f-af18-43fe1e57c957

[root@rhos0 ~(refarch_member)]# neutron router-interface-add route1  \
$subnet_id
Added interface 1cc125ca-60d7-4c1e-9076-be988fb008e0 to router route1.
```

12. Create security group rules to allow traffic to instances in this security group. **Table 4.3.1.1: Allowed Ports by Role** lists the ports required by OpenStack and Sahara.

```
[root@rhos0 ~(refarch_member)]# default_id=$(neutron security-group-list |
awk ' /default/ { print $2 } ')

[root@rhos0 ~(refarch_member)]# echo -e "default id: $default_id"
default id: 34ad791b-9708-4f93-822b-a493e12d62ae

[root@rhos0 ~(refarch_member)]# neutron security-group-rule-create \
--direction ingress --protocol icmp  $default_id
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | ef41f407-0628-4921-a837-f4dec8728f36 |
| port_range_max    |                                      |
| port_range_min    |                                      |
| protocol          | icmp                                 |
| remote_group_id   |                                      |
```

```
| remote_ip_prefix  |                                      |
| security_group_id | 34ad791b-9708-4f93-822b-a493e12d62ae |
| tenant_id         | 7f367eecfda24aabb15f401df8cd1b70     |
+-------------------+--------------------------------------+

[root@rhos0 ~(refarch_member)]# neutron security-group-rule-create \
--direction ingress --protocol tcp --port_range_min 22 --port_range_max 22 \
$default_id
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 4449ec26-8295-43e4-8ad5-4fb356517621 |
| port_range_max    | 22                                   |
| port_range_min    | 22                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 34ad791b-9708-4f93-822b-a493e12d62ae |
| tenant_id         | 7f367eecfda24aabb15f401df8cd1b70     |
+-------------------+--------------------------------------+

[root@rhos0 ~(refarch_member)]# neutron security-group-rule-create \
--direction ingress --protocol tcp --port_range_min 80 --port_range_max 80 \
$default_id
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | da8745d2-d82b-41c4-a7c0-dda4777d5e26 |
| port_range_max    | 80                                   |
| port_range_min    | 80                                   |
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 34ad791b-9708-4f93-822b-a493e12d62ae |
| tenant_id         | 7f367eecfda24aabb15f401df8cd1b70     |
+-------------------+--------------------------------------+

[root@rhos0 ~(refarch_member)]# neutron security-group-rule-create \
--direction ingress --protocol tcp --port_range_min 873 \
--port_range_max 873  $default_id
Created a new security_group_rule:
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| direction         | ingress                              |
| ethertype         | IPv4                                 |
| id                | 84640ca3-bf2e-4e80-92d2-0500f9c0dcee |
| port_range_max    | 873                                  |
| port_range_min    | 873                                  |
```

```
| protocol          | tcp                                  |
| remote_group_id   |                                      |
| remote_ip_prefix  |                                      |
| security_group_id | 34ad791b-9708-4f93-822b-a493e12d62ae |
| tenant_id         | 7f367eecfda24aabb15f401df8cd1b70     |
+-------------------+--------------------------------------+
```

NOTE: Command output is shown only for the first few ports and protocols. The remainder are truncated for brevity. Repeat the **`neutron-securitygroup-rule-create`** command for all ports.

13. Display the new security group rules.

```
[root@rhos0 ~(refarch_member)]# neutron security-group-show $default_id
+---------------------------------------+------+-------------------+--------+
| Field                 | Value
+---------------------------------------+------+-------------------+--------+
| description           | default
| id                    | 34ad791b-9708-4f93-822b-a493e12d62ae
| name                  | default
| security_group_rules | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 8442,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 8040, "ethertype": "IPv4", "id": "02a799f8-a92f-40cb-aa28-
645db67bd0a5"}                                              |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 10020,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 10020, "ethertype": "IPv4", "id": "068b07a3-52ea-4c3e-
a82d-825d7790cedc"}                                             |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50030,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50030, "ethertype": "IPv4", "id": "141be0ab-28cb-4184-
926d-90e2df294bdc"}                                             |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 22,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 22, "ethertype": "IPv4", "id": "4449ec26-8295-43e4-8ad5-
4fb356517621"}                                                   |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50010,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50010, "ethertype": "IPv4", "id": "4f342eb2-e50d-40c6-
8b2a-aacc9624f074"}                                             |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50075,
```

```
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50075, "ethertype": "IPv4", "id": "507b23bd-cb69-437b-
a9e5-a70f2494d516"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50070,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50070, "ethertype": "IPv4", "id": "51aa45c1-b3be-4243-
bd9b-5ec9af81be88"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 8088,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 8088, "ethertype": "IPv4", "id": "5980bca2-b71b-47d2-aa12-
33ccd5acf0e6"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50090,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50090, "ethertype": "IPv4", "id": "5cb68dc0-5f16-44c1-
83af-6b81d762442d"}                                |
|                       | {"remote_group_id": null, "direction": "egress",
"remote_ip_prefix": null, "protocol": null, "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": null,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": null, "ethertype": "IPv6", "id": "625aa91d-f583-427e-b1b3-
7d83fd2db61f"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 8080,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 8080, "ethertype": "IPv4", "id": "6de68bf0-3dcd-47fc-bbab-
88ac9a8b4af9"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 19888,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 19888, "ethertype": "IPv4", "id": "7753d48a-3849-4c42-
aef9-d18f95c08ee5"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 8033,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 8030, "ethertype": "IPv4", "id": "7def9bf2-9332-492a-bcbf-
d84ef880fb79"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 8021,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 8020, "ethertype": "IPv4", "id": "8095eeeb-6fc2-41b7-b947-
aa43b91c69b9"}                                |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 6002,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
```

```
"port_range_min": 6000, "ethertype": "IPv4", "id": "82afc1a4-0167-4e70-9e1a-
13f355fbdb0d"}                                     |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 873,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 873, "ethertype": "IPv4", "id": "84640ca3-bf2e-4e80-92d2-
0500f9c0dcee"}                                       |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50020,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50020, "ethertype": "IPv4", "id": "88fce240-06f2-4a78-
94a0-5b6547594a3a"}                                   |
|                       | {"remote_group_id": "34ad791b-9708-4f93-822b-
a493e12d62ae", "direction": "ingress", "remote_ip_prefix": null, "protocol":
null, "tenant_id": "7f367eecfda24aabb15f401df8cd1b70", "port_range_max":
null, "security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": null, "ethertype": "IPv6", "id": "9da8fcc4-47dc-46f1-bb1d-
15d1bffc60a6"} |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 11000,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 11000, "ethertype": "IPv4", "id": "b4994f43-1534-4c49-
94bc-274d68e8ca74"}                                   |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 50060,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 50060, "ethertype": "IPv4", "id": "bac001bc-cbb3-4928-
bf38-3ac70715fed1"}                                   |
|                       | {"remote_group_id": null, "direction": "egress",
"remote_ip_prefix": null, "protocol": null, "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": null,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": null, "ethertype": "IPv4", "id": "c7b19bf5-030a-490e-850b-
79877f38d06c"}                                       |
|                       | {"remote_group_id": "34ad791b-9708-4f93-822b-
a493e12d62ae", "direction": "ingress", "remote_ip_prefix": null, "protocol":
null, "tenant_id": "7f367eecfda24aabb15f401df8cd1b70", "port_range_max":
null, "security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": null, "ethertype": "IPv4", "id": "ce2361a1-99b7-460a-9a0d-
57f48d5f233a"} |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 9000,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 9000, "ethertype": "IPv4", "id": "d7439aab-fa0f-4f57-88b2-
b67e36bd8e37"}                                     |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "tcp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": 80,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": 80, "ethertype": "IPv4", "id": "da8745d2-d82b-41c4-a7c0-
```

```
dda4777d5e26"}                                      |
|                       | {"remote_group_id": null, "direction": "ingress",
"remote_ip_prefix": null, "protocol": "icmp", "tenant_id":
"7f367eecfda24aabb15f401df8cd1b70", "port_range_max": null,
"security_group_id": "34ad791b-9708-4f93-822b-a493e12d62ae",
"port_range_min": null, "ethertype": "IPv4", "id": "ef41f407-0628-4921-a837-
f4dec8728f36"}                                 |
| tenant_id             | 7f367eecfda24aabb15f401df8cd1b70
+---------------------------------------+------+-------------------+--------+
```

14. Set the external network gateway on the router to the external network.

```
[root@rhos0 ~(refarch_member)]# route_id=$(neutron router-list | awk '
/route1/ { print $2 }')

[root@rhos0 ~(refarch_member)]# echo -e "route_id: $route_id"
route_id: 643ebbfe-2186-4272-ae10-46444e5d128b

[root@rhos0 ~(refarch_member)]# ext_net_id=$(neutron net-list | awk '
/ext_net/ { print $2 } ')

[root@rhos0 ~(refarch_member)]# echo -e "ext_net_id: $ext_net_id"
ext_net_id: 97680250-36e7-49be-bdcd-44c6ac86b0d1

[root@rhos0 ~(refarch_member)]# neutron router-gateway-set $route_id
$ext_net_id
Set gateway for router 643ebbfe-2186-4272-ae10-46444e5d128b
```

15. Create a key pair.

```
[root@rhos0 ~(refarch_member)]# nova keypair-add refarchkp >
/root/refarchkp.pem

[root@rhos0 ~(refarch_member)]# chmod 600 /root/refarchkp.pem
```

5.4 Install Sahara

1. Open port 8386 for Sahara API communication on the cloud controller.

NOTE: This example uses **iptables** due to the underlying Puppet installer. The equivalent **firewalld** command may also be used with Red Hat Enterprise Linux 7.

```
[root@rhos0 ~]# iptables -I INPUT 1 -p tcp --dport 8386 -j ACCEPT

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

2. Install the Sahara packages.

```
[root@rhos0 ~]# yum install -y -q openstack-sahara python-saharaclient \
```

```
python-django-sahara
```

3. Create the Sahara database. In this example the OpenStack state database is used for Sahara. It is possible to use an alternate database.

```
[root@rhos0 ~]# mysql -u root --password=redhat -e "CREATE DATABASE sahara;"

[root@rhos0 ~]# mysql -u root --password=redhat -e "GRANT ALL ON sahara.* \
TO  'sahara'@'%' IDENTIFIED BY 'saharatest';"

[root@rhos0 ~]# openstack-config --set /etc/sahara/sahara.conf database \
connection mysql://sahara:saharatest@172.16.2.100/sahara

[root@rhos0 ~]# sahara-db-manage --config-file /etc/sahara/sahara.conf \
upgrade head
INFO  [alembic.migration] Context impl MySQLImpl.
INFO  [alembic.migration] Will assume non-transactional DDL.
INFO  [alembic.migration] Running upgrade None -> 001
INFO  [alembic.migration] Running upgrade 001 -> 002
INFO  [alembic.migration] Running upgrade 002 -> 003
INFO  [alembic.migration] Running upgrade 003 -> 004
INFO  [alembic.migration] Running upgrade 004 -> 005
INFO  [alembic.migration] Running upgrade 005 -> 006

[root@rhos0 ~]# mysql -u sahara --password=saharatest sahara -e \
"show tables;"
+----------------------+
| Tables_in_sahara     |
+----------------------+
| alembic_version      |
| cluster_templates    |
| clusters             |
| data_sources         |
| instances            |
| job_binaries         |
| job_binary_internal  |
| job_executions       |
| jobs                 |
| libs_association     |
| mains_association    |
| node_group_templates |
| node_groups          |
| templates_relations  |
+----------------------+
```

4. Source *keystonerc_admin* and create the Sahara user in the ***services*** tenant.

```
[root@rhos0 ~]# source /root/keystonerc_admin

[root@rhos0 ~(openstack_admin)]# keystone user-create --name sahara \
--pass saharatest
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
```

```
|  email   |                                  |
| enabled  |               True               |
|    id    | 4eba1510090940b1907f773a3e7da03c |
|   name   |              sahara              |
| username |              sahara              |
+----------+----------------------------------+

[root@rhos0 ~(openstack_admin)]# keystone user-role-add --user sahara \
--role admin --tenant services
```

5. Create a Keystone service for Sahara of type *data_processing*.

```
[root@rhos0 ~(openstack_admin)]# keystone service-create –name=sahara \
--type=data_processing --description="Sahara data processing"
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |      Sahara data processing      |
|   enabled   |               True               |
|      id     | eb31dcbae5c2473db54860edc03d92c6 |
|     name    |              sahara              |
|     type    |         data_processing          |
+-------------+----------------------------------+
```

6. Create a Keystone service endpoint for Sahara on port 8386.

```
[root@rhos0 ~(openstack_admin)]# keystone endpoint-create --service sahara \
--publicurl "http://10.19.137.100:8386/v1.1/%(tenant_id)s" \
--adminurl "http://172.16.2.100:8386/v1.1/%(tenant_id)s" \
--internalurl "http://172.16.2.100:8386/v1.1/%(tenant_id)s"
+-------------+---------------------------------------------+
|   Property  |                    Value                    |
+-------------+---------------------------------------------+
|   adminurl  | http://172.16.2.100:8386/v1.1/%(tenant_id)s  |
|      id     |       8e47d0477d144b42bf5b24170d030006       |
| internalurl | http://172.16.2.100:8386/v1.1/%(tenant_id)s  |
|  publicurl  | http://10.19.137.100:8386/v1.1/%(tenant_id)s |
|    region   |                  regionOne                   |
|  service_id |       eb31dcbae5c2473db54860edc03d92c6       |
+-------------+---------------------------------------------+
```

7. Configure the Sahara service to use Keystone authentication and Neutron networking.

```
[root@rhos0 ~(openstack_admin)]# openstack-config \
–set /etc/sahara/sahara.conf DEFAULT os_auth_host 172.16.2.100

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT os_auth_port 5000

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT os_admin_tenant_name services

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT os_admin_username sahara
```

```
[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT os_admin_password saharatest

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT use_neutron true

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT log_dir /var/log/sahara

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT verbose true

[root@rhos0 ~(openstack_admin)]# openstack-config \
--set /etc/sahara/sahara.conf DEFAULT debug false
```

8. Start Sahara on the cloud controller.

```
[root@rhos0 ~(openstack_admin)]# systemctl start \
openstack-sahara-api.service

[root@rhos0 ~(openstack_admin)]# systemctl enable \
openstack-sahara-api.service
ln -s '/usr/lib/systemd/system/openstack-sahara-api.service'
'/etc/systemd/system/multi-user.target.wants/openstack-sahara-api.service'

[root@rhos0 ~(openstack_admin)]# systemctl status \
openstack-sahara-api.service
openstack-sahara-api.service - Sahara API Server
   Loaded: loaded (/usr/lib/systemd/system/openstack-sahara-api.service;
enabled)
   Active: active (running) since Wed 2014-10-01 13:33:00 CDT; 129ms ago
 Main PID: 22011 (sahara-api)
   CGroup: /system.slice/openstack-sahara-api.service
           \u2514\u250022011 /usr/bin/python2 /usr/bin/sahara-api --config-
file /etc/sahara/sahara.conf

Oct 01 13:33:00 rhos0.cloud.lab.eng.bos.redhat.com systemd[1]: Started
Sahara API Server.
```

9. List the installed Sahara plugins.

```
[root@rhos0 ~(openstack_admin)]# sahara plugin-list
+---------------------------------------------------------------------------+
| name    | versions     | title
-+--------------------------------------------------------------------------+
| vanilla | 1.2.1, 2.3.0 | Vanilla Apache Hadoop
| hdp     | 1.3.2, 2.0.6 | Hortonworks Data Platform
| idh     | 2.5.1, 3.0.2 | Intel(R) Distribution for Apache Hadoop* Software
+---------+--------------+--------------------------------------------------+
```

5.4.1 Configure the Sahara Dashboard plugin

1. Add *saharadashboard* to the **INSTALLED_APPS** section of the Horizon dashboard

configuration file.

```
[root@rhos0 ~(openstack_admin)]# perl -p -i.orig -e 's/INSTALLED_APPS =
[[]\n/INSTALLED_APPS = [\n    \047saharadashboard\047,\n/'
/usr/share/openstack-dashboard/openstack_dashboard/settings.py
```

2. Add Sahara to the OpenStack Dashboard.

```
[root@rhos0 ~(openstack_admin)]# perl -p -i -e
's/\047router\047/\047router\047, \047sahara\047/' /usr/share/openstack-
dashboard/openstack_dashboard/settings.py
```

3. Configure Sahara to use Neutron networking in the *local_settings* file. Sahara uses Nova networking by default.

```
[root@rhos0 ~(openstack_admin)]# echo -e "SAHARA_USE_NEUTRON =
True\nSAHARA_URL =
'http://172.16.2.100:8386/v1.1'\nAUTO_ASSIGNMENT_ENABLED=False" >>
/etc/openstack-dashboard/local_settings
```

4. Restart HTTPD for the changes to take effect.

```
[root@rhos0 ~(openstack_admin)]# systemctl restart httpd.service

[root@rhos0 ~(openstack_admin)]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)
   Active: active (running) since Wed 2014-10-01 13:33:12 CDT; 8ms ago
  Process: 22059 ExecStop=/bin/kill -WINCH ${MAINPID} (code=exited,
status=0/SUCCESS)

 Main PID: 22067 (httpd)
   Status: "Processing requests..."
   CGroup: /system.slice/httpd.service
           \u251c\u250022067 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022068 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022069 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022070 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022071 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022072 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022073 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022074 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022075 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022076 /usr/sbin/httpd -DFOREGROUND
           \u251c\u250022077 /usr/sbin/httpd -DFOREGROUND
           \u2514\u250022078 /usr/sbin/httpd -DFOREGROUND

Oct 01 13:33:12 rhos0.cloud.lab.eng.bos.redhat.com httpd[22067]: AH00548:
NameVirtualHost has no effect and will be removed in the next release
/etc/httpd/conf/ports.conf:7

Oct 01 13:33:12 rhos0.cloud.lab.eng.bos.redhat.com systemd[1]: Started The
Apache HTTP Server.
```

5.5 Configure Sahara

1. Configure Sahara on the cloud controller as the tenant user.

```
[root@rhos0 ~(openstack_admin)]# source /root/keystonerc_refarch

[root@rhos0 ~(refarch_member)]# env | grep OS_
OS_PASSWORD=refarch
OS_AUTH_URL=http://10.19.137.100:35357/v2.0/
OS_USERNAME=refarch
OS_TENANT_NAME=refarch-tenant
```

2. Export Hadoop environment variables.

```
[root@rhos0 ~(refarch_member)]# sahara plugin-list
+---------+-------------+-------------------------------------------------+
| name    | versions    | title                                           |
+---------+-------------+-------------------------------------------------+
| vanilla | 1.2.1, 2.3.0 | Vanilla Apache Hadoop
| hdp     | 1.3.2, 2.0.6 | Hortonworks Data Platform
| idh     | 2.5.1, 3.0.2 | Intel(R) Distribution for Apache Hadoop* Software
+---------+-------------+-------------------------------------------------+

[root@rhos0 ~(refarch_member)]# export PLUGIN_NAME=vanilla

[root@rhos0 ~(refarch_member)]# export PLUGIN_VERSION=2.3.0

[root@rhos0 ~(refarch_member)]# export MASTER_FLAVOR=m1.medium

[root@rhos0 ~(refarch_member)]# export WORKER_FLAVOR=m1.medium

[root@rhos0 ~(refarch_member)]# export WORKER_COUNT=3
```

3. Export OpenStack environment variables.

```
[root@rhos0 ~(refarch_member)]# export MANAGEMENT_NETWORK=net1

[root@rhos0 ~(refarch_member)]# EXT_NET_ID=$(nova net-list | awk ' /ext_net/
{ print $2 } ')

[root@rhos0 ~(refarch_member)]# echo $EXT_NET_ID
072a9419-ecbe-4403-a278-1032735b1de5

[root@rhos0 ~(refarch_member)]# export KEYPAIR=refarchkp
```

5.6 Import Glance Images for Hadoop

1. Create or download a Glance image for Hadoop. The image used in this example was created with Sahara Image Elements[2].

   NOTE: Links to community-maintained images can be found at

2 http://docs.openstack.org/developer/sahara/userdoc/diskimagebuilder.html

http://docs.openstack.org/developer/sahara/userdoc/vanilla_plugin.html

2. Create the Glance images for Hadoop.

```
[root@rhos0 ~(refarch_member)]# glance image-create \
--name centos64_hadoop2  --is-public true --disk-format qcow2 \
--container-format bare \
--file /pub/projects/images/centos_sahara_vanilla_hadoop_2_latest.qcow2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | e0c4da81a754a20a402cd5a6139edbb3     |
| container_format | bare                                 |
| created_at       | 2014-10-01T18:38:30                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | centos64_hadoop2                     |
| owner            | 7f367eecfda24aabb15f401df8cd1b70     |
| protected        | False                                |
| size             | 1144717312                           |
| status           | active                               |
| updated_at       | 2014-10-01T18:38:40                  |
| virtual_size     | None                                 |
+------------------+--------------------------------------+
```

3. Ensure the Glance image is **Active**.

```
[root@rhos0 ~(refarch_member)]# glance image-list
+--------------------------------------+------------------+-------------+--+
| ID                                   | Name             | Disk Format |
Container Format | Size       | Status |
+--------------------------------------+------------------+-------------+--+
| cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a | centos64_hadoop2 | qcow2       |
bare             | 1144717312 | active |
+--------------------------------------+------------------+-------------+--+
```

4. Register the image with Sahara.

```
[root@rhos0 ~(refarch_member)]# export IMAGE_ID=$(glance image-list | awk '
/centos64_hadoop2/ { print $ 2 } ')

[root@rhos0 ~(refarch_member)]# echo -e "Image ID: $IMAGE_ID"
Image ID: cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a

[root@rhos0 ~(refarch_member)]# sahara image-register --id $IMAGE_ID
--username cloud-user

[root@rhos0 ~(refarch_member)]# sahara image-add-tag --id $IMAGE_ID --tag
```

```
$PLUGIN_NAME

[root@rhos0 ~(refarch_member)]# sahara image-add-tag --id $IMAGE_ID --tag
$PLUGIN_VERSION

[root@rhos0 ~(refarch_member)]# sahara image-list
+-------------------+---------------------------------------+------------+---+
| name              | id                                    | username   |
tags             | description |
+-------------------+---------------------------------------+------------+---+
| centos64_hadoop2 | cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a | cloud-user |
2.3.0, vanilla | None        |
+-------------------+---------------------------------------+------------+---+
```

5.7 Define Templates

1. Node and cluster templates are reusable objects that define clusters. Define environment variables common to the templates.

```
[root@rhos0 ~(refarch_member)]# export MASTER_FLAVOR_ID=$(nova flavor-show
$MASTER_FLAVOR | awk ' / id / {print $4}')

[root@rhos0 ~(refarch_member)]# export WORKER_FLAVOR_ID=$(nova flavor-show
$WORKER_FLAVOR | awk ' / id / {print $4}')

[root@rhos0 ~(refarch_member)]# export MANAGEMENT_NETWORK_ID=$(neutron net-
show $MANAGEMENT_NETWORK | awk ' / id / {print $4}')
```

5.7.1 Define Master Group Template

1. Create a JSON file named */root/master_group_template.json* that defines the master node of the cluster.

```
[root@rhos0 ~(refarch_member)]# tee /root/master_group_template.json << EOF
{
 "plugin_name": "$PLUGIN_NAME",
 "node_processes": [ "namenode", "resourcemanager", "oozie", "historyserver"
],
 "flavor_id": "$MASTER_FLAVOR_ID",
 "floating_ip_pool": "$EXT_NET_ID",
 "hadoop_version": "$PLUGIN_VERSION",
 "name": "master"
 }
EOF
{
"plugin_name": "vanilla",
"node_processes": [ "namenode", "resourcemanager", "oozie", "historyserver"
],
"flavor_id": "3",
"floating_ip_pool": "97680250-36e7-49be-bdcd-44c6ac86b0d1",
"hadoop_version": "2.3.0",
"name": "master"
}
```

2. Create the ***master*** group template.

```
[root@rhos0 ~(refarch_member)]# sahara node-group-template-create \
--json /root/master_group_template.json | awk '/\| id/ { print $4 } '
cbd6ced1-0fb3-4c32-a1a4-70a5dd07e730

[root@rhos0 ~(refarch_member)]# sahara node-group-template-show --name
master
+---------------------+---------------------------------------------------+
| Property            | Value                                             |
+---------------------+---------------------------------------------------+
| created_at          | 2014-10-01 18:39:16                               |
| description         | None                                              |
| flavor_id           | 3                                                 |
| floating_ip_pool    | 97680250-36e7-49be-bdcd-44c6ac86b0d1              |
| hadoop_version      | 2.3.0                                             |
| id                  | cbd6ced1-0fb3-4c32-a1a4-70a5dd07e730              |
| image_id            | None                                              |
| name                | master                                            |
| node_configs        | {}                                                |
| node_processes      | namenode, resourcemanager, oozie, historyserver |
| plugin_name         | vanilla                                           |
| tenant_id           | 7f367eecfda24aabb15f401df8cd1b70                  |
| updated_at          | None                                              |
| volume_mount_prefix | /volumes/disk                                     |
| volumes_per_node    | 0                                                 |
| volumes_size        | 0                                                 |
+---------------------+---------------------------------------------------+
```

3. Set the ***MASTER_TEMPLATE_ID*** environment variable.

```
[root@rhos0 ~(refarch_member)]# export MASTER_TEMPLATE_ID=$(sahara node-
group-template-show --name master | awk '/\| id/ { print $4 } ')
```

5.7.2 Define the Worker Group Template

1. Create a JSON file named */root/worker_group_template.json* that defines the worker nodes of the cluster.

```
[root@rhos0 ~(refarch_member)]# tee /root/worker_group_template.json << EOF
{
"plugin_name": "$PLUGIN_NAME",
"node_processes": [ "datanode", "nodemanager" ],
"flavor_id": "$WORKER_FLAVOR_ID",
"floating_ip_pool": "$EXT_NET_ID",
"hadoop_version": "$PLUGIN_VERSION",
"name": "worker"
}
EOF
{
"plugin_name": "vanilla",
"node_processes": [ "datanode", "nodemanager" ],
"flavor_id": "3",
"floating_ip_pool": "97680250-36e7-49be-bdcd-44c6ac86b0d1",
"hadoop_version": "2.3.0",
```

```
"name": "worker"
}
```

2. Create the ***worker*** group template.

```
[root@rhos0 ~(refarch_member)]# sahara node-group-template-create --json
/root/worker_group_template.json | awk '/\| id/ { print $4 } '
5ec60092-c579-43ba-a49c-14222d4381a3

[root@rhos0 ~(refarch_member)]# sahara node-group-template-show --name
worker
+---------------------+--------------------------------------+
| Property            | Value                                |
+---------------------+--------------------------------------+
| created_at          | 2014-10-01 18:39:17                  |
| description         | None                                 |
| flavor_id           | 3                                    |
| floating_ip_pool    | 97680250-36e7-49be-bdcd-44c6ac86b0d1 |
| hadoop_version      | 2.3.0                                |
| id                  | 5ec60092-c579-43ba-a49c-14222d4381a3 |
| image_id            | None                                 |
| name                | worker                               |
| node_configs        | {}                                   |
| node_processes      | datanode, nodemanager                |
| plugin_name         | vanilla                              |
| tenant_id           | 7f367eecfda24aabb15f401df8cd1b70     |
| updated_at          | None                                 |
| volume_mount_prefix | /volumes/disk                        |
| volumes_per_node    | 0                                    |
| volumes_size        | 0                                    |
+---------------------+--------------------------------------+
```

3. Set the ***WORKER_TEMPLATE_ID*** environment variable.

```
[root@rhos0 ~(refarch_member)]# export WORKER_TEMPLATE_ID=$(sahara node-
group-template-show --name worker | awk '/\| id/ { print $4 } ')
```

4. List the node templates.

```
[root@rhos0 ~(refarch_member)]# sahara node-group-template-list
+--------+--------------------------------------+-------------+-------------+
| name   | id                                   | plugin_name |
node_processes                                  | description |
+--------+--------------------------------------+-------------+-------------+
| worker | 5ec60092-c579-43ba-a49c-14222d4381a3 | vanilla     | datanode,
nodemanager                                  | None         |
| master | cbd6ced1-0fb3-4c32-a1a4-70a5dd07e730 | vanilla     | namenode,
resourcemanager, oozie, historyserver | None         |
+--------+--------------------------------------+-------------+-------------+
```

5.7.3 Define Cluster Templates

1. Create a JSON file named */root/cluster_template.json* that defines a cluster of one

master node and three worker nodes using the node group templates.

```
[root@rhos0 ~(refarch_member)]# tee /root/cluster_template.json << EOF
{
"plugin_name": "$PLUGIN_NAME",
"node_groups": [
        { "count": 1,
        "name": "master",
        "node_group_template_id": "$MASTER_TEMPLATE_ID" },
        { "count": $WORKER_COUNT,
        "name": "worker",
        "node_group_template_id": "$WORKER_TEMPLATE_ID" } ],
"hadoop_version": "$PLUGIN_VERSION",
"name": "cluster"
}
EOF
{
"plugin_name": "vanilla",
"node_groups": [
        { "count": 1,
        "name": "master",
        "node_group_template_id": "cbd6ced1-0fb3-4c32-a1a4-70a5dd07e730" },
        { "count": 3,
        "name": "worker",
        "node_group_template_id": "5ec60092-c579-43ba-a49c-14222d4381a3" }
],
"hadoop_version": "2.3.0",
"name": "cluster"
}
```

2. Create a cluster template from the JSON definition.

```
[root@rhos0 ~(refarch_member)]# sahara cluster-template-create --json
/root/cluster_template.json
+----------------------------+--------------------------------------+
| Property                   | Value                                |
+----------------------------+--------------------------------------+
| anti_affinity              | []                                   |
| cluster_configs            | {}                                   |
| created_at                 | 2014-10-01 18:39:33                  |
| default_image_id           | None                                 |
| description                | None                                 |
| hadoop_version             | 2.3.0                                |
| id                         | e629139b-9c61-46a9-92e2-0b77eff60bc1 |
| name                       | cluster                              |
| neutron_management_network | None                                 |
| node_groups                | master: 1, worker: 3                 |
| plugin_name                | vanilla                              |
| tenant_id                  | 7f367eecfda24aabb15f401df8cd1b70     |
| updated_at                 | None                                 |
+----------------------------+--------------------------------------+
```

3. List the available cluster templates.

```
[root@rhos0 ~(refarch_member)]# sahara cluster-template-list
+---------+--------------------------------------+-------------+-----------+
| name    | id                                   | plugin_name | node_groups
| description |
+---------+--------------------------------------+-------------+-----------+
| cluster | e629139b-9c61-46a9-92e2-0b77eff60bc1 | vanilla     | master: 1,
worker: 3 | None        |
+---------+--------------------------------------+-------------+-----------+
```

4. Set the ***CLUSTER_TEMPLATE_ID*** environment variable.

```
[root@rhos0 ~(refarch_member)]# export CLUSTER_TEMPLATE_ID=$(sahara cluster-
template-show --name cluster  | awk '/\| id/ { print $4 } ')
```

5.8 Launch a Cluster via the Command Line

1. Create a cluster definition file named */root/cluster.json*.

```
[root@rhos0 ~(refarch_member)]# tee /root/cluster.json << EOF
{
"cluster_template_id": "$CLUSTER_TEMPLATE_ID",
"default_image_id": "$IMAGE_ID",
"hadoop_version": "$PLUGIN_VERSION",
"name": "cluster-instance-$(date +%s)",
"plugin_name": "$PLUGIN_NAME",
"user_keypair_id": "$KEYPAIR",
"neutron_management_network": "$MANAGEMENT_NETWORK_ID"
}
EOF
{
"cluster_template_id": "e629139b-9c61-46a9-92e2-0b77eff60bc1",
"default_image_id": "cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a",
"hadoop_version": "2.3.0",
"name": "cluster-instance-1412188774",
"plugin_name": "vanilla",
"user_keypair_id": "refarchkp",
"neutron_management_network": "0c0bc665-c992-4594-b7c5-7b94690e94e3"
}
```

2. Launch a cluster from the template via the Sahara command line.

```
[root@rhos0 ~(refarch_member)]# sahara cluster-create --json
/root/cluster.json
+----------------------------+-------------------------------------------+
| Property                   | Value
+----------------------------+-------------------------------------------+
| anti_affinity              | []
| cluster_configs            | {}
| cluster_template_id        | e629139b-9c61-46a9-92e2-0b77eff60bc1
| created_at                 | 2014-10-01 18:39:36
| default_image_id           | cb5bcb96-e8ec-4ff6-a04c-1eaec7c9c25a
| description                | None
| hadoop_version             | 2.3.0
| id                         | d9f3b71d-19fd-4f00-81c1-d4e9b0c4b0b9
```

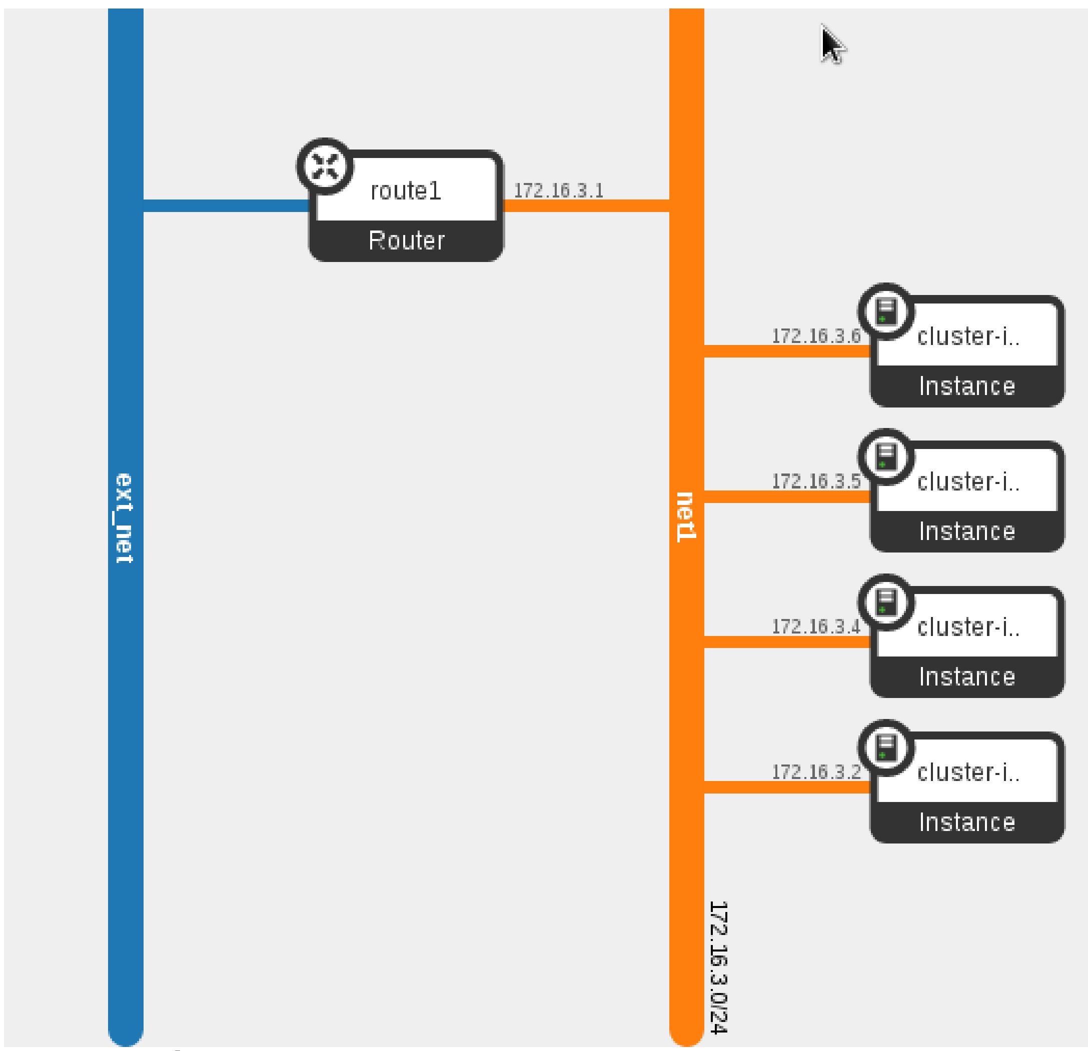
```
| info                       | {}
| is_transient               | False
| management_public_key      | ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDZZ3jzzxA |
|                            |
fVGbyMr7OvrPOIKF+OnVZcWpy6/Qr7cGNKI+8l4yjXXLXTs |
|                            |
MnIhk+WXxbym3hIkaSqnZdKIidG0HhIJsQJ5CA/8vVcwUpl |
|                            |
edrknNKtEHu1cPAJtXUQoim3hrAG0B49gd47n+3SzetUNh6 |
|                            |
f/L974hd9YMAEd60ahGzsD+i+brhTfvTPO+yCgOVCFhwylO |
|                            |
2Y8UXK9AQh3H6kfYaB1dFyLSxBdMSMQYNw92wPAYcq47Jpj |
|                            |
dmqVzIImPuArcN00EuIKyOD1ih5upDzxdzzf9xhqmwDstC9 |
|                            |
wu/T+3Ig1g24dTSMxNxn5GM3fRFtr3nb1hdrjiPVBY4/QCP |
|                            | oQ7x Generated by Sahara
| name                       | cluster-instance-1412188774
| neutron_management_network | 0c0bc665-c992-4594-b7c5-7b94690e94e3
| node_groups                | [{u'count': 3, u'name': u'worker',
u'image_id': |
|                            | None, u'volume_mount_prefix':
u'/volumes/disk', |
|                            | u'created_at': u'2014-10-01 18:39:36',
|                            | u'updated_at': u'2014-10-01 18:39:36',
|                            | u'floating_ip_pool': u'97680250-36e7-49be-
bdcd- |
|                            | 44c6ac86b0d1', u'instances': [],
|                            | u'volumes_size': 0, u'node_configs': {},
|                            | u'node_group_template_id': u'5ec60092-c579
|                            | -43ba-a49c-14222d4381a3',
u'volumes_per_node':  |
|                            | 0, u'node_processes': [u'datanode',
|                            | u'nodemanager'], u'flavor_id': u'3'},
|                            | {u'count': 1, u'name': u'master',
u'image_id':  |
|                            | None, u'volume_mount_prefix':
u'/volumes/disk', |
|                            | u'created_at': u'2014-10-01 18:39:36',
|                            | u'updated_at': u'2014-10-01 18:39:36',
|                            | u'floating_ip_pool': u'97680250-36e7-49be-
bdcd- |
|                            | 44c6ac86b0d1', u'instances': [],
|                            | u'volumes_size': 0, u'node_configs': {},
|                            | u'node_group_template_id':
|                            | u'cbd6ced1-0fb3-4c32-a1a4-70a5dd07e730',
|                            | u'volumes_per_node': 0, u'node_processes':
|                            | [u'namenode', u'resourcemanager', u'oozie',
|                            | u'historyserver'], u'flavor_id': u'3'}]
| plugin_name                | vanilla
| status                     | Validating
| status_description         |
| tenant_id                  | 7f367eecfda24aabb15f401df8cd1b70
```

```
| trust_id                    | None
| updated_at                  | 2014-10-01 18:39:37
| user_keypair_id             | refarchkp
+-----------------------------+-----------------------------------------------+
```

3. Verify the cluster is in an **Active** state.

```
[root@rhos0 ~(refarch_member)]# sahara cluster-list
+-----------------------------+----------------------+--------+------------+
| name                        | id                                   |
status | node_count |
+-----------------------------+----------------------+--------+------------+
| cluster-instance-1412188774 | d9f3b71d-19fd-4f00-81c1-d4e9b0c4b0b9 |
Active | 4          |
+-----------------------------+----------------------+--------+------------+
```

4. After the cluster is deployed successfully, the network and cluster layout should match the topology outlined in **Graphic 5.8.1: Sahara Cluster Topology**.



Graphic 5.8.1: Sahara Cluster Topology

6 Validate and Test Sahara

This section describes how to explore and validate the cluster installation by connecting to the master node and running a test MapReduce job.

6.1 Explore the Cluster

1. Find the floating IP address of the master node. In this example it is 10.19.137.115.

```
[root@rhos0 ~(refarch_member)]# nova list
+--------+----------------------------------------+--------+--------+------+
| ID                                       | Name
| Status | Task State | Power State | Networks                        |
+--------+----------------------------------------+--------+--------+------+
| b0fccab4-36d5-4bf0-aeea-472ac518d427 | cluster-instance-1412188774-master-
001 | ACTIVE | -           | Running     | net1=172.16.3.6, 10.19.137.115 |
| bf139b98-48ae-4b40-9c4a-25752dcf2d5b | cluster-instance-1412188774-worker-
001 | ACTIVE | -           | Running     | net1=172.16.3.2, 10.19.137.113 |
| 223a10aa-ca88-4da7-8457-86b114c0bf8a | cluster-instance-1412188774-worker-
002 | ACTIVE | -           | Running     | net1=172.16.3.4, 10.19.137.112 |
| 20965761-9c86-4f2a-9ac7-9c62697b9009 | cluster-instance-1412188774-worker-
003 | ACTIVE | -           | Running     | net1=172.16.3.5, 10.19.137.114 |
+--------+----------------------------------------+--------+--------+------+
```

2. SSH to the master node by floating IP address. By default, the image allows SSH login from the cloud-user account with authentication via the key pair specified in section **5.8 Launch a Cluster via the Command Line**.

```
[root@rhos0 ~(refarch_member)]# ssh -l cloud-user -i /root/refarchkp.pem
10.19.137.115
```

3. After login, switch to the root user.

```
[cloud-user@cluster-instance-1412188774-master-001 ~]$ sudo su
```

4. View the running Java processes on the master node. In this example the master is running ***NameNode***, ***ResourceManager***, and the ***JobHistoryServer***.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# jps
1701 ResourceManager
3101 Bootstrap
1591 NameNode
2197 JobHistoryServer
15818 Jps
```

5. View the environment variables related to Hadoop.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# env | grep -i \
hadoop
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/bin:/usr/java/jdk1.7.0_51/bin:
/opt/hadoop/bin:/opt/hadoop/sbin
```

```
HADOOP_HDFS_HOME=/opt/hadoop
HADOOP_COMMON_HOME=/opt/hadoop
HADOOP_YARN_HOME=/opt/hadoop
HADOOP_MAPRED_HOME=/opt/hadoop
```

6. View the unique HDFS version ID, cluster ID, and blockpool ID.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# grep ID \
/mnt/hdfs/namenode/current/VERSION

namespaceID=1547179929
clusterID=CID-9f3887af-81f0-4b49-a547-324d41dcef05
blockpoolID=BP-678581577-172.16.3.6-1412188926657
```

7. View the ***NameNode*** and ***secondaryNameNode*** configuration. In this example both run on the master node.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# su hadoop -c "hdfs
getconf -namenodes"
cluster-instance-1412188774-master-001

[root@cluster-instance-1412188774-master-001 cloud-user]# su hadoop -c "hdfs
getconf -secondaryNameNodes"
0.0.0.0
```

8. Run a report on the HDFS with **dfsadmin**.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# su - hadoop -c
"hdfs dfsadmin -report"

Configured Capacity: 31671791616 (29.50 GB)
Present Capacity: 22913183692 (21.34 GB)
DFS Remaining: 22676205568 (21.12 GB)
DFS Used: 236978124 (226.00 MB)
DFS Used%: 1.03%
Under replicated blocks: 0
Blocks with corrupt replicas: 0
Missing blocks: 0

-------------------------------------------------
Datanodes available: 3 (3 total, 0 dead)

Live datanodes:
Name: 172.16.3.5:50010 (cluster-instance-1412188774-worker-003.novalocal)
Hostname: cluster-instance-1412188774-worker-003.novalocal
Decommission Status : Normal
Configured Capacity: 10557263872 (9.83 GB)
DFS Used: 78992708 (75.33 MB)
Non DFS Used: 2919656124 (2.72 GB)
DFS Remaining: 7558615040 (7.04 GB)
DFS Used%: 0.75%
DFS Remaining%: 71.60%
Configured Cache Capacity: 0 (0 B)
```

```
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Last contact: Wed Oct 01 22:47:25 MSK 2014

Name: 172.16.3.4:50010 (cluster-instance-1412188774-worker-002.novalocal)
Hostname: cluster-instance-1412188774-worker-002.novalocal
Decommission Status : Normal
Configured Capacity: 10557263872 (9.83 GB)
DFS Used: 78992708 (75.33 MB)
Non DFS Used: 2919541436 (2.72 GB)
DFS Remaining: 7558729728 (7.04 GB)
DFS Used%: 0.75%
DFS Remaining%: 71.60%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%

Last contact: Wed Oct 01 22:47:25 MSK 2014

Name: 172.16.3.2:50010 (cluster-instance-1412188774-worker-001.novalocal)
Hostname: cluster-instance-1412188774-worker-001.novalocal
Decommission Status : Normal
Configured Capacity: 10557263872 (9.83 GB)
DFS Used: 78992708 (75.33 MB)
Non DFS Used: 2919410364 (2.72 GB)
DFS Remaining: 7558860800 (7.04 GB)
DFS Used%: 0.75%
DFS Remaining%: 71.60%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Last contact: Wed Oct 01 22:47:25 MSK 2014
```

9. List the nodes seen by Yarn.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# su - hadoop -c
"yarn node --list"

Total Nodes:3
         Node-Id           Node-State  Node-Http-Address      Number-of-
Running-Containers
cluster-instance-1412188774-worker-001.novalocal:39779          RUNNING
    cluster-instance-1412188774-worker-001.novalocal:8042
0
cluster-instance-1412188774-worker-003.novalocal:47762          RUNNING
    cluster-instance-1412188774-worker-003.novalocal:8042
0
cluster-instance-1412188774-worker-002.novalocal:50125          RUNNING
```

```
    cluster-instance-1412188774-worker-002.novalocal:8042
0
```

10. Run a file system check.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# su - hadoop -c
"hdfs fsck /"

Connecting to namenode via http://cluster-instance-1412188774-master-
001:50070
FSCK started by hadoop (auth:SIMPLE) from /172.16.3.6 for path / at Wed Oct
01 22:47:30 MSK 2014
.............................................................................
 Status: HEALTHY
 Total size:    78355058 B
 Total dirs:    21
 Total files:   106
 Total symlinks:            0
 Total blocks (validated): 106 (avg. block size 739198 B)
 Minimally replicated blocks:    106 (100.0 %)
 Over-replicated blocks:    0 (0.0 %)
 Under-replicated blocks:  0 (0.0 %)
 Mis-replicated blocks:          0 (0.0 %)
 Default replication factor:     3
 Average block replication:      3.0
 Corrupt blocks:            0
 Missing replicas:          0 (0.0 %)
 Number of data-nodes:           3
 Number of racks:           1

FSCK ended at Wed Oct 01 22:47:30 MSK 2014 in 30 milliseconds

The filesystem under path '/' is HEALTHY
```

6.2 Run a Test Hadoop Job from the Command Line

1. Switch to the Hadoop user.

```
[root@cluster-instance-1412188774-master-001 cloud-user]# su - hadoop
```

2. Execute the ***Pi*** MapReduce example to verify Hadoop functionality.

```
[hadoop@cluster-instance-1412188774-master-001 ~]$ cd \
/opt/hadoop-2.3.0/share/hadoop/mapreduce/

[hadoop@cluster-instance-1412188774-master-001 mapreduce]$ hadoop jar
hadoop-mapreduce-examples-2.3.0.jar pi 10 100

Number of Maps  = 10
Samples per Map = 100
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
```

```
Wrote input for Map #4
Wrote input for Map #5
Wrote input for Map #6
Wrote input for Map #7
Wrote input for Map #8
Wrote input for Map #9
Starting Job
14/10/01 22:45:35 INFO client.RMProxy: Connecting to ResourceManager at
cluster-instance-1412188774-master-001/172.16.3.6:8032

14/10/01 22:45:35 INFO input.FileInputFormat: Total input paths to process :
10
14/10/01 22:45:35 INFO mapreduce.JobSubmitter: number of splits:10
14/10/01 22:45:36 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1412188934308_0001
14/10/01 22:45:36 INFO impl.YarnClientImpl: Submitted application
application_1412188934308_0001
14/10/01 22:45:36 INFO mapreduce.Job: The url to track the job:
http://cluster-instance-1412188774-master-
001:8088/proxy/application_1412188934308_0001/
14/10/01 22:45:36 INFO mapreduce.Job: Running job: job_1412188934308_0001
14/10/01 22:45:43 INFO mapreduce.Job: Job job_1412188934308_0001 running in
uber mode : false
14/10/01 22:45:43 INFO mapreduce.Job:  map 0% reduce 0%
14/10/01 22:45:56 INFO mapreduce.Job:  map 40% reduce 0%
14/10/01 22:45:59 INFO mapreduce.Job:  map 100% reduce 0%
14/10/01 22:46:02 INFO mapreduce.Job:  map 100% reduce 100%
14/10/01 22:46:02 INFO mapreduce.Job: Job job_1412188934308_0001 completed
successfully

14/10/01 22:46:02 INFO mapreduce.Job: Counters: 49
     File System Counters
           FILE: Number of bytes read=226
           FILE: Number of bytes written=965162
           FILE: Number of read operations=0
           FILE: Number of large read operations=0
           FILE: Number of write operations=0
           HDFS: Number of bytes read=2940
           HDFS: Number of bytes written=215
           HDFS: Number of read operations=43
           HDFS: Number of large read operations=0
           HDFS: Number of write operations=3

     Job Counters
           Launched map tasks=10
           Launched reduce tasks=1
           Data-local map tasks=10
           Total time spent by all maps in occupied slots (ms)=116399
           Total time spent by all reduces in occupied slots (ms)=3007
           Total time spent by all map tasks (ms)=116399
           Total time spent by all reduce tasks (ms)=3007
           Total vcore-seconds taken by all map tasks=116399
           Total vcore-seconds taken by all reduce tasks=3007
           Total megabyte-seconds taken by all map tasks=119192576
           Total megabyte-seconds taken by all reduce tasks=3079168
```

```
    Map-Reduce Framework
          Map input records=10
          Map output records=20
          Map output bytes=180
          Map output materialized bytes=280
          Input split bytes=1760
          Combine input records=0
          Combine output records=0
          Reduce input groups=2
          Reduce shuffle bytes=280
          Reduce input records=20
          Reduce output records=0
          Spilled Records=40
          Shuffled Maps =10
          Failed Shuffles=0
          Merged Map outputs=10
          GC time elapsed (ms)=713
          CPU time spent (ms)=3620
          Physical memory (bytes) snapshot=2438373376
          Virtual memory (bytes) snapshot=9596039168
          Total committed heap usage (bytes)=1890058240

    Shuffle Errors
          BAD_ID=0
          CONNECTION=0
          IO_ERROR=0
          WRONG_LENGTH=0
          WRONG_MAP=0
          WRONG_REDUCE=0

    File Input Format Counters
          Bytes Read=1180
    File Output Format Counters
          Bytes Written=97

Job Finished in 27.109 seconds
Estimated value of Pi is 3.14800000000000000000
```

6.3 Run a Test EDP Job Through Sahara

In the previous section a user submitted a Hadoop job via the command line. Users may also define and submit reusable jobs directly via Sahara. This section demonstrates how to define a job template and submit a Pig job that uses Swift as a data store.

1. Source *keystonerc_admin* to set OpenStack admin environment variables.

```
[root@rhos0 ~]# source /root/keystonerc_admin

[root@rhos0 ~(openstack_admin)]# env | grep OS_
OS_PASSWORD=redhat
OS_AUTH_URL=http://172.16.2.100:35357/v2.0/
OS_USERNAME=admin
OS_TENANT_NAME=admin
```

2. Add the ***SwiftOperator*** role to the tenant user.

```
[root@rhos0 ~(openstack_admin)]# keystone user-role-add --user refarch
--role SwiftOperator --tenant refarch-tenant
```

3. Switch to the tenant user environment.

```
[root@rhos0 ~(openstack_admin)]# source /root/keystonerc_refarch

[root@rhos0 ~(refarch_member)]# env | grep OS_
OS_PASSWORD=refarch
OS_AUTH_URL=http://10.19.137.100:35357/v2.0/
OS_USERNAME=refarch
OS_TENANT_NAME=refarch-tenant
```

4. Copy the job source files to a temporary directory.

   NOTE: At the time of writing, this code example is downloaded to the cloud controller's local directory ***$SRC_DIR*** from here:
   https://github.com/openstack/sahara/tree/master/etc/edp-examples/pig-job

```
[root@rhos0 ~(refarch_member)]# mkdir -p /tmp/pig-test

[root@rhos0 ~(refarch_member)]# cp -r $SRC_DIR/* /tmp/pig-test/

[root@rhos0 ~(refarch_member)]# cd /tmp/pig-test
```

5. Create a Swift container and Sahara data sources for input and output.

```
[root@rhos0 pig-test(refarch_member)]# swift post container4

[root@rhos0 pig-test(refarch_member)]# sahara data-source-create --name
rainput --type swift --url swift://container4.sahara/input --user refarch
--password refarch
+-------------+--------------------------------------+
| Property    | Value                                |
+-------------+--------------------------------------+
| created_at  | 2014-10-01 18:49:47.600212           |
| description |                                      |
| id          | 42695c77-c701-4306-a88a-9e2efc97c041 |
| name        | rainput                              |
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70     |
| type        | swift                                |
| url         | swift://container4.sahara/input      |
+-------------+--------------------------------------+

[root@rhos0 pig-test(refarch_member)]# sahara data-source-create --name
raoutput --type swift --url swift://container4.sahara/output --user refarch
--password refarch
+-------------+--------------------------------------+
| Property    | Value                                |
```

```
+-------------+-------------------------------------+
| created_at  | 2014-10-01 18:49:48.110776          |
| description |                                     |
| id          | 1d4e6f5f-f125-46ae-8e0b-1d3643f1b6b4 |
| name        | raoutput                            |
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70    |
| type        | swift                               |
| url         | swift://container4.sahara/output    |
+-------------+-------------------------------------+
```

6. Upload the input files to the Sahara input data store.

```
[root@rhos0 pig-test(refarch_member)]# for UPFILES in input example.pig
udf.jar; do swift upload container4 ${UPFILES}; done
input
example.pig
udf.jar
```

7. Create the Sahara job binaries.

```
[root@rhos0 pig-test(refarch_member)]# for BINFILE in example.pig udf.jar;
do sahara job-binary-create --name ${BINFILE} --url
swift://container4.sahara/${BINFILE} --user refarch --password refarch; done
+-------------+--------------------------------------+
| Property    | Value                                |
+-------------+--------------------------------------+
| created_at  | 2014-10-01 18:49:50.704531           |
| description |                                      |
| id          | 272f899a-9f3e-461e-b79e-57949fdef8c2 |
| name        | example.pig                          |
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70     |
| url         | swift://container4.sahara/example.pig |
+-------------+--------------------------------------+

+-------------+-------------------------------------+
| Property    | Value                               |
+-------------+-------------------------------------+
| created_at  | 2014-10-01 18:49:51.187247          |
| description |                                     |
| id          | 2aff7aa7-e5cd-43e6-9a76-cdabd242f8b3 |
| name        | udf.jar                             |
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70    |
| url         | swift://container4.sahara/udf.jar   |
+-------------+-------------------------------------+
```

8. Get the Sahara job binary IDs.

```
[root@rhos0 pig-test(refarch_member)]# JOB_MAIN_ID=$(sahara job-binary-list
| awk '/ example.pig / {print $2}')

[root@rhos0 pig-test(refarch_member)]# JOB_LIB_ID=$(sahara job-binary-list |
awk '/ udf.jar / {print $2}')

[root@rhos0 pig-test(refarch_member)]# echo $JOB_MAIN_ID
```

```
272f899a-9f3e-461e-b79e-57949fdef8c2

[root@rhos0 pig-test(refarch_member)]# echo $JOB_LIB_ID
2aff7aa7-e5cd-43e6-9a76-cdabd242f8b3
```

9. Create a job template of type Pig that includes the job binaries.

```
[root@rhos0 pig-test(refarch_member)]# sahara job-template-create --name
pig-test --type Pig --main ${JOB_MAIN_ID} --lib ${JOB_LIB_ID}
+-------------+---------------------------------------------------------------+
| created_at  | 2014-10-01 18:50:18.222140
| description |
| id          | de8ccecb-6b29-4f61-99a5-e3ec434945e7
| libs        | [{u'description': u'', u'url':
u'swift://container4.sahara/udf.jar', u'tenant_id':
u'7f367eecfda24aabb15f401df8cd1b70', u'created_at': u'2014-10-01 18:49:51',
u'updated_at': None, u'id': u'2aff7aa7-e5cd-43e6-9a76-cdabd242f8b3',
u'name': u'udf.jar'}]         |
| mains       | [{u'description': u'', u'url':
u'swift://container4.sahara/example.pig', u'tenant_id':
u'7f367eecfda24aabb15f401df8cd1b70', u'created_at': u'2014-10-01 18:49:50',
u'updated_at': None, u'id': u'272f899a-9f3e-461e-b79e-57949fdef8c2',
u'name': u'example.pig'}] |
| name        | pig-test
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70
| type        | Pig
+-------------+---------------------------------------------------------------+
```

10. Set the ID for the job template to an environment variable.

```
[root@rhos0 pig-test(refarch_member)]# JOB_TEMPLATE_ID=$(sahara  job-
template-list | awk '/ pig-test / {print $2}')

[root@rhos0 pig-test(refarch_member)]# echo $JOB_TEMPLATE_ID
de8ccecb-6b29-4f61-99a5-e3ec434945e7
```

11. Get the cluster ID of the previously launched cluster.

```
[root@rhos0 pig-test(refarch_member)]# CLUSTER_ID=$(sahara  cluster-list |
tail -n +4 | awk '{print $4}' | head -n 1)

[root@rhos0 pig-test(refarch_member)]# echo $CLUSTER_ID
d9f3b71d-19fd-4f00-81c1-d4e9b0c4b0b9
```

12. Set the job input data store ID to an environment variable.

```
[root@rhos0 pig-test(refarch_member)]# DATA_INPUT_ID=$(sahara  data-source-
list | awk '/^\| rainput / {print $4}')

[root@rhos0 pig-test(refarch_member)]# echo $DATA_INPUT_ID
42695c77-c701-4306-a88a-9e2efc97c041
```

13. Set the job output ID to an environment variable.

```
[root@rhos0 pig-test(refarch_member)]# DATA_OUTPUT_ID=$(sahara  data-source-
list | awk '/^\| raoutput / {print $4}')

[root@rhos0 pig-test(refarch_member)]# echo $DATA_OUTPUT_ID
1d4e6f5f-f125-46ae-8e0b-1d3643f1b6b4
```

14. Launch a job from the job template to the cluster, specifying the input and output data stores.

```
[root@rhos0 pig-test(refarch_member)]# sahara  job-create --job-template $
{JOB_TEMPLATE_ID} --cluster ${CLUSTER_ID} --input-data ${DATA_INPUT_ID}
--output-data ${DATA_OUTPUT_ID}
+-------------+--------------------------------------------+
| Property    | Value                                      |
+-------------+--------------------------------------------+
| cluster_id  | d9f3b71d-19fd-4f00-81c1-d4e9b0c4b0b9       |
| created_at  | 2014-10-01 18:50:20.797040                 |
| id          | 933a2067-5477-42a2-a270-fa0ddf9ce93e       |
| input_id    | 42695c77-c701-4306-a88a-9e2efc97c041       |
| job_configs | {u'configs': {}, u'args': [], u'params': {}} |
| job_id      | de8ccecb-6b29-4f61-99a5-e3ec434945e7       |
| output_id   | 1d4e6f5f-f125-46ae-8e0b-1d3643f1b6b4       |
| status      | Pending                                    |
| tenant_id   | 7f367eecfda24aabb15f401df8cd1b70           |
+-------------+--------------------------------------------+
```

15. Verify that the job ran successfully.

```
[root@rhos0 pig-test(refarch_member)]# sahara job-list
+--------------------------------------+----------------------+-----------+
| id                                   | cluster_id
| status    |
+--------------------------------------+----------------------+-----------+
| 933a2067-5477-42a2-a270-fa0ddf9ce93e | d9f3b71d-19fd-4f00-81c1-
d4e9b0c4b0b9 | SUCCEEDED |
+--------------------------------------+----------------------+-----------+

[root@rhos0 pig-test(refarch_member)]# sahara job-show --id 933a2067-5477-
42a2-a270-fa0ddf9ce93e
+--------------+--------------------------------------------+
| Property     | Value                                      |
+--------------+--------------------------------------------+
| cluster_id   | d9f3b71d-19fd-4f00-81c1-d4e9b0c4b0b9       |
| created_at   | 2014-10-01 18:50:20                        |
| end_time     | 2014-10-01T13:51:45                        |
| id           | 933a2067-5477-42a2-a270-fa0ddf9ce93e       |
| input_id     | 42695c77-c701-4306-a88a-9e2efc97c041       |
| job_configs  | {u'configs': {}, u'args': [], u'params': {}} |
| job_id       | de8ccecb-6b29-4f61-99a5-e3ec434945e7       |
| oozie_job_id | 0000000-141001224319614-oozie-hado-W       |
| output_id    | 1d4e6f5f-f125-46ae-8e0b-1d3643f1b6b4       |
| progress     | None                                       |
| return_code  | None                                       |
| start_time   | 2014-10-01T13:50:51                        |
```

```
| status        | SUCCEEDED                                     |
| tenant_id     | 7f367eecfda24aabb15f401df8cd1b70              |
| updated_at    | 2014-10-01 18:51:45                           |
+---------------+-----------------------------------------------+
```

16. List the contents of the Swift container.

```
[root@rhos0 ~(refarch_member)]# swift list container4
example.pig
input
output
output/_SUCCESS
output/part-m-00000
udf.jar
```

17. Download the output to verify the spaces are removed.

```
[root@rhos0 ~(refarch_member)]# swift download container4 output/part-m-
00000
output/part-m-00000 [auth 0.204s, headers 0.253s, total 0.253s, 0.001 MB/s]

[root@rhos0 pig-test(refarch_member)]# diff -y input output/part-m-00000
   pomegranate                                          |     pomegranate
      banana                                            |     banana
  apple                                                 |     apple
   lychee                                               |     lychee
```

18. Connect to the cluster to view the Oozie workflow logs.

```
[root@rhos0 ~(refarch_member)]# ssh -l cloud-user -i /root/refarchkp.pem
10.19.137.115

[cloud-user@cluster-instance-1412188774-master-001 ~]$ sudo su

[hadoop@cluster-instance-1412188774-master-001 ~]$ /opt/oozie/bin/oozie jobs
-oozie http://localhost:11000/oozie
Job ID                                   App Name     Status    User
Group     Started                 Ended
----------------------------------------------------------------------------
0000000-141001224319614-oozie-hado-W     job-wf       SUCCEEDED hadoop    -
2014-10-01 18:50 GMT    2014-10-01 18:51 GMT
----------------------------------------------------------------------------
```

19. View the Oozie job definition.

```
[hadoop@cluster-instance-1412188774-master-001 ~]$ /opt/oozie/bin/oozie job
-oozie http://localhost:11000/oozie -definition 0000000-141001224319614-
oozie-hado-W

<?xml version="1.0" ?>
<workflow-app name="job-wf" xmlns="uri:oozie:workflow:0.2">
  <start to="job-node"/>
  <action name="job-node">
```

```
    <pig>
      <job-tracker>${jobTracker}</job-tracker>
      <name-node>${nameNode}</name-node>
      <configuration>
        <property>
          <name>fs.swift.service.sahara.password</name>
          <value>refarch</value>
        </property>
        <property>
          <name>fs.swift.service.sahara.username</name>
          <value>refarch</value>
        </property>
      </configuration>
      <script>example.pig</script>
      <param>INPUT=swift://container4.sahara/input</param>
      <param>OUTPUT=swift://container4.sahara/output</param>
    </pig>
    <ok to="end"/>
    <error to="fail"/>
  </action>
  <kill name="fail">
    <message>Workflow failed, error message[$
{wf:errorMessage(wf:lastErrorNode())}]</message>
  </kill>
  <end name="end"/>
</workflow-app>
```

7 Conclusion

Sahara enables OpenStack users to deploy virtual Hadoop clusters and run data processing jobs on them. It combines the power of large scale data analytics with the convenience and flexibility of OpenStack, enabling data scientists to spend their time solving data science problems rather than managing software and infrastructure. This reference architecture describes how to implement Sahara in a RHEL OSP 5 environment and submit a test job. It also describes Sahara’s high level architecture and provides an overview of how Sahara fits into the OpenStack ecosystem.

This reference architecture focuses on the single user to small group use case. Sahara is particularly useful for developing and testing data processing applications before running them on a production cluster. Sahara is also useful for elastic data processing, where additional data processing resources are deployed as needed in order to accommodate the occasional workload that is too large for the production cluster.

At the time of writing Sahara is a technology preview. Sahara is expected to be fully supported in the Juno release. Subsequent reference architectures describe additional use cases such as integrating OpenStack Sahara with an existing production Hadoop cluster and running a production cluster within OpenStack.

Appendix A: References

1. What does a "Technology Preview" feature mean? **https://access.redhat.com/solutions/21101**
2. Red Hat Enterprise Linux 7 installation instructions: **https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Installation_Guide/index.html**
3. Red Hat Enterprise Linux OpenStack Platform 5 deployment guide: **https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/5/html/Installer_and_Foreman_Guide/index.html**
4. Building images for Sahara Plugins **http://docs.openstack.org/developer/sahara/userdoc/diskimagebuilder.html**

Appendix B: Revision History

Revision 1.0	Thursday October 9, 2014	Jacob Liberman

- Incorporated Services review feedback
- Final layout and spacing
- Verified internal document links

Revision 0.4	Wednesday October 1, 2014	Jacob Liberman

- Finalized content
- Incorporated Systems Engineering review feedback

Revision 0.3	Monday September 29, 2014	Jacob Liberman

- Incorporated Product Engineering review feedback

Revision 0.2	Friday September 5, 2014	Jacob Liberman

- Table of contents
- Executive summary
- Ported to template version 11

Appendix C: Host Group YAML Output

C.1 Cloud Controller


```
---
classes:
  quickstack::neutron::controller:
    admin_email: admin@cloud.lab.eng.bos.redhat.com
    admin_password: redhat
    amqp_ca: /etc/ipa/ca.crt
    amqp_cert: /etc/pki/tls/certs/PRIV_HOST-amqp.crt
    amqp_host: 172.16.2.100
    amqp_key: /etc/pki/tls/private/PRIV_HOST-amqp.key
    amqp_nssdb_password: redhat
    amqp_password: redhat
    amqp_provider: rabbitmq
    amqp_username: openstack
    ceilometer_metering_secret: redhat
    ceilometer_user_password: redhat
    cinder_backend_eqlx: 'false'
    cinder_backend_eqlx_name:
    - eqlx_backend
    cinder_backend_gluster: 'false'
    cinder_backend_gluster_name: glusterfs_backend
    cinder_backend_iscsi: 'false'
    cinder_backend_iscsi_name: iscsi_backend
    cinder_backend_nfs: 'True'
    cinder_backend_nfs_name: nfs_backend
    cinder_backend_rbd: 'false'
    cinder_backend_rbd_name: rbd_backend
    cinder_db_password: redhat
    cinder_eqlx_chap_login:
    - chapadmin
    cinder_eqlx_chap_password:
    - redhat
    cinder_eqlx_group_name:
    - group-0
    cinder_eqlx_pool:
    - default
    cinder_eqlx_use_chap:
    - 'false'
    cinder_gluster_shares:
    - 192.168.0.4:/cinder -o backup-volfile-servers=192.168.0.5
    cinder_multiple_backends: 'false'
    cinder_nfs_mount_options: ''
    cinder_nfs_shares:
    - 10.19.137.120:/cinder
    cinder_rbd_ceph_conf: /etc/ceph/ceph.conf
    cinder_rbd_flatten_volume_from_snapshot: 'false'
    cinder_rbd_max_clone_depth: '5'
    cinder_rbd_pool: volumes
    cinder_rbd_secret_uuid: ''
```

```
    cinder_rbd_user: volumes
    cinder_san_ip:
    - 192.168.124.11
    cinder_san_login:
    - grpadmin
    cinder_san_password:
    - redhat
    cinder_san_thin_provision:
    - 'false'
    cinder_user_password: redhat
    cisco_nexus_plugin:
neutron.plugins.cisco.nexus.cisco_nexus_plugin_v2.NexusPlugin
    cisco_vswitch_plugin:
neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2
    controller_admin_host: 172.16.2.100
    controller_priv_host: 172.16.2.100
    controller_pub_host: 10.19.137.100
    enable_tunneling: 'False'
    freeipa: 'false'
    glance_backend: file
    glance_db_password: redhat
    glance_rbd_store_pool: images
    glance_rbd_store_user: images
    glance_user_password: redhat
    heat_auth_encrypt_key: redhat
    heat_cfn: 'true'
    heat_cloudwatch: 'false'
    heat_db_password: redhat
    heat_user_password: redhat
    horizon_ca: /etc/ipa/ca.crt
    horizon_cert: /etc/pki/tls/certs/PUB_HOST-horizon.crt
    horizon_key: /etc/pki/tls/private/PUB_HOST-horizon.key
    horizon_secret_key: redhat
    keystonerc: 'true'
    keystone_admin_token: redhat
    keystone_db_password: redhat
    ml2_firewall_driver:
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
    ml2_flat_networks:
    - ! '*'
    ml2_mechanism_drivers:
    - openvswitch
    - l2population
    ml2_network_vlan_ranges:
    - physint:1000:1010
    - physext
    ml2_security_group: 'true'
    ml2_tenant_network_types:
    - flat
    - vlan
    ml2_tunnel_id_ranges:
    - 20:100
    ml2_type_drivers:
    - flat
    - vlan
```

```
    - local
    ml2_vni_ranges:
    - 10:100
    ml2_vxlan_group: 224.0.0.1
    mysql_ca: /etc/ipa/ca.crt
    mysql_cert: /etc/pki/tls/certs/PRIV_HOST-mysql.crt
    mysql_host: 172.16.2.100
    mysql_key: /etc/pki/tls/private/PRIV_HOST-mysql.key
    mysql_root_password: redhat
    neutron_core_plugin: neutron.plugins.ml2.plugin.Ml2Plugin
    neutron_db_password: redhat
    neutron_metadata_proxy_secret: redhat
    neutron_user_password: redhat
    nexus_config: {}
    nexus_credentials: []
    nova_db_password: redhat
    nova_default_floating_pool: nova
    nova_user_password: redhat
    ovs_vlan_ranges: ! '''physint:1000:1010,physext'''
    provider_vlan_auto_create: 'false'
    provider_vlan_auto_trunk: 'false'
    ssl: 'false'
    swift_admin_password: redhat
    swift_ringserver_ip: 172.31.139.100
    swift_shared_secret: redhat
    swift_storage_device: device1
    swift_storage_ips:
    - 172.31.139.107
    - 172.31.139.108
    - 172.31.139.109
    tenant_network_type: vlan
    tunnel_id_ranges: 1:1000
    verbose: 'true'
```

C.2 Neutron Networker

```
---
classes:
  quickstack::neutron::networker:
    amqp_host: 172.16.2.100
    amqp_password: redhat
    amqp_provider: rabbitmq
    amqp_username: openstack
    controller_priv_host: 172.16.2.100
    enable_tunneling: 'False'
    external_network_bridge: ''
    fixed_network_range: 172.16.3.0/24
    mysql_ca: /etc/ipa/ca.crt
    mysql_host: 172.16.2.100
    neutron_db_password: redhat
    neutron_metadata_proxy_secret: redhat
    neutron_user_password: redhat
    nova_db_password: redhat
    nova_user_password: redhat
    ovs_bridge_mappings:
```

```
    - physint:br-enp21
    - physext:br-eno2
    ovs_bridge_uplinks:
    - br-enp21:enp21
    - br-eno2:eno2
    ovs_l2_population: 'True'
    ovs_tunnel_iface: eth6
    ovs_tunnel_network: ''
    ovs_tunnel_types:
    - vxlan
    ovs_vlan_ranges: ! '''physint:1000:1010,physext'''
    ovs_vxlan_udp_port: '4789'
    ssl: 'false'
    tenant_network_type: vlan
    tunnel_id_ranges: 1:1000
    verbose: 'true'
```

C.3 Compute Node

```
---
classes:
  quickstack::neutron::compute:
    admin_password: redhat
    amqp_host: 172.16.2.100
    amqp_password: redhat
    amqp_port: '5672'
    amqp_provider: rabbitmq
    amqp_ssl_port: '5671'
    amqp_username: openstack
    auth_host: 172.16.2.100
    ceilometer: 'False'
    ceilometer_metering_secret: redhat
    ceilometer_user_password: redhat
    cinder_backend_gluster: 'false'
    cinder_backend_nfs: 'True'
    cinder_backend_rbd: 'false'
    enable_tunneling: 'False'
    glance_host: 172.16.2.100
    libvirt_images_rbd_ceph_conf: /etc/ceph/ceph.conf
    libvirt_images_rbd_pool: volumes
    libvirt_images_type: rbd
    libvirt_inject_key: 'false'
    libvirt_inject_password: 'false'
    mysql_ca: /etc/ipa/ca.crt
    mysql_host: 172.16.2.100
    neutron_db_password: redhat
    neutron_host: 172.16.2.100
    neutron_user_password: redhat
    nova_db_password: redhat
    nova_host: 172.16.2.100
    nova_user_password: redhat
    ovs_bridge_mappings:
    - physint:br-enp21
    - physext:br-eno2
    ovs_bridge_uplinks:
```

```
    - br-enp21:enp21
    - br-eno2:eno2
    ovs_l2_population: 'True'
    ovs_tunnel_iface: eth6
    ovs_tunnel_network: ''
    ovs_tunnel_types:
    - vxlan
    ovs_vlan_ranges: ! '''physint:1000:1010,physext'''
    ovs_vxlan_udp_port: '4789'
    private_iface: ''
    private_ip: ''
    private_network: ''
    rbd_secret_uuid: ''
    rbd_user: volumes
    ssl: 'false'
    tenant_network_type: vlan
    tunnel_id_ranges: 1:1000
    verbose: 'true'
```

C.4 Swift Storage Server

```
---
classes:
  quickstack::swift::storage:
    swift_all_ips:
    - 172.31.139.101
    - 172.31.139.102
    - 172.31.139.103
    - 172.31.139.104
    - 172.31.139.105
    - 172.31.139.107
    - 172.31.139.108
    - 172.31.139.109
    - 172.31.139.100
    swift_ext4_device: /dev/sdb1
    swift_local_interface: p1p2
    swift_local_network: ''
    swift_loopback: 'False'
    swift_ring_server: 172.31.139.100
    swift_shared_secret: redhat
```