RED HAT CONSULTING

RED HAT ENTERPRISE LINUX OPENSTACK PLATFORM POC

ENGAGEMENT REPORT
PREPARED FOR: MOC

CONFIDENTIAL

TABLE OF CONTENTS

1 PREFACE

1.1 Confidentiality, Copyright, and Disclaimer

1.2 About This Document

This is a confidential document between Red Hat, Inc. and Cerner detailing a Red Hat OpenStack Install engagement. It is intended for technical staff responsible for implementing and maintaining a Red Hat OpenStack private cloud.

1.3 Additional Background and Related Documents

Other related material specific to this solution:

- Red Hat Documentation at https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/ including:
    1. Getting Started Guide
    2. Installation and Configuration Guide
- Reference Architecture related to this solution at https://access.redhat.com/site/articles/reference-architecture.
    1. Deploying and Using Red Hat OpenStack (RHOS) 2.1
    2. Deploying and Using Red Hat OpenStack (RHOS) 3.0
- Ceph installation documentation can be found at http://ceph.com/docs/master/start/

1.4 Terms

The table below provides a glossary of the terms and acronyms used within this document.

Term	Description
Calamari	Graphical Interface to manage Ceph
Ceilometer	OpenStack Telemetry (Metering) service
Cinder	OpenStack Persistent block storage service
Cloud Controller	Master node in OpenStack configuration usually providing public Nova API, Keystone identity services, Glance image services and scheduling.
Compute node	OpenStack node running virtual machines
Flat network	Network without VLANs
Foreman	Provisioning system shipped with Red Hat OpenStack product
Glance	OpenStack Image service
Heat	OpenStack Orchestration service
Horizon	OpenStack Dashboard (HTTP) service
Keystone	OpenStack Identity service
MON	Ceph Monitor process. Monitors cluster health
Network node	OpenStack node providing network management and connectivity for compute nodes
Neutron	OpenStack networking services
Nova	OpenStack Compute service
Object Storage node	Storage node providing Swift services
OSD	Ceph Object Storage Device. A physical disk in a storage node
OVS	Open vSwitch
Packstack	Red Hat provisioning tool for OpenStack
Provider network	Representation of a physical network in OpenStack configuration that makes it available for cloud tenants.
Puppet	System configuration tool used by Packstack and Foreman for OpenStack deployment
Quantum	The former name of the Neutron OpenStack networking services
RHEL	Red Hat Enterprise Linux
RHEL OSP	Red Hat Enterprise Linux OpenStack Platform
Swift	Object Storage services
Tenant network	A network used by tenants of OpenStack. (Also called private networks)

Table 1-1: Terms Defined

2 ENGAGEMENT SUMMARY

This implementation was the start of building out capabilities for the MOC. It began by implementing Red Hat Satellite 6. This allows a central point for provisioning, configuration management, and content distribution. It builds off the previous capability of the OpenStack Foreman installer. During this work we also implemented a Ceph storage deployment with Inktank Ceph Enterprise 1.2 (Firefly). Ceph was deployed in a 3 node cluster with object storage devices and monitors co-located. In addition Calamari was deployed for management and a single Rados Gateway was deployed for Swift integration. While initially just installed on virtual machines this will eventually migrate to physical hardware. Ceph will provide the basis of OpenStack Nova, Cinder, Glance, and Swift storage. Once done with this we customized puppet modules to enable a consolidated controller/networker node. We also added the capability to deploy Sahara, LbaaS, FwaaS, and VPNaaS. Finally we used this infrastructure to deploy RHEL OSP 5 (Icehouse) in a non-HA configuration. This included one consolidated controller/network nodeand 13 compute nodes. We saved 2 nodes for further testing. Vxlan was used as the network overlay to facilitate tenant networks. This was used due to the fact VLAN would not work with our given network configuration.

2.1 Contact Information

Role	Purpose	Red Hat Assignment	Contact Info
Domain Architect	Design, architecture, and implementation	Jon Jozwiak	763.218.2530 jjozwiak@redhat.com
OpenStack Consultant	Design, architecture, and implementation	David Critch	dcritch@redhat.com
OpenStack Consultant	Design, architecture, and implementation	Tolga Ozaltin	tozaltin@redhat.com

3 TECHNICAL IMPLEMENTATION

3.1 Technical Details

3.1.1 Hardware Type and Location

Host	Platform	Notes
controller	RHEL7	OpenStack Controller / Networker
compute01	RHEL7	OpenStack Compute Node
compute02	RHEL7	OpenStack Compute Node
compute03	RHEL7	OpenStack Compute Node
compute04	RHEL7	OpenStack Compute Node
compute05	RHEL7	OpenStack Compute Node
compute06	RHEL7	OpenStack Compute Node
compute07	RHEL7	OpenStack Compute Node
compute08	RHEL7	OpenStack Compute Node
compute09	RHEL7	OpenStack Compute Node
compute10	RHEL7	OpenStack Compute Node
compute11	RHEL7	OpenStack Compute Node
compute12	RHEL7	OpenStack Compute Node
compute13	RHEL7	OpenStack Compute Node
compute14	RHEL7	UNUSED – Reserved for testing
compute15	RHEL7	UNUSED – Reserved for testing
eos-ctl01	CentOS	KVM Host
Moc-sat	RHEL7 Guest	Satellite 6
Moc-calamari	RHEL7 Guest	Calamari / Ceph Repository
Moc-ceph01	RHEL7 Guest	Ceph Node 1
Moc-ceph02	RHEL7 Guest	Ceph Node 2
Moc-ceph03	RHEL7 Guest	Ceph Node 3
Moc-mon	RHEL6 Guest	Monitoring Host
Moc-rgw	RHEL7 Guest	Ceph Rados Gateway

3.1.2 IP Address Details

Host Name	Mgmt IP	Public IP	Storage IP	Role
controller	10.31.27.207			OpenStack Controller

Host Name	Mgmt IP	Public IP	Storage IP	Role
compute01	10.31.27.208			OpenStack Compute Node
compute02	10.31.27.209			OpenStack Compute Node
compute03	10.31.27.210			OpenStack Compute Node
compute04	10.31.27.211			OpenStack Compute Node
compute05	10.31.27.212			OpenStack Compute Node
compute06	10.31.27.213			OpenStack Compute Node
compute07	10.31.27.214			OpenStack Compute Node
compute08	10.31.27.215			OpenStack Compute Node
compute09	10.31.27.216			OpenStack Compute Node
compute10	10.31.27.217			OpenStack Compute Node
compute11	10.31.27.218			OpenStack Compute Node
compute12	10.31.27.219			OpenStack Compute Node
compute13	10.31.27.220			OpenStack Compute Node
compute14	10.31.27.221			UNUSED
compute15	10.31.27.222			UNUSED
eos-ctl01	10.31.27.11			KVM Host
Moc-sat	10.31.27.201			Satellite 6
Moc-calamari	10.31.27.205			Calamari / Ceph Repository
Moc-ceph01	10.31.27.202			Ceph Node 1
Moc-ceph02	10.31.27.203			Ceph Node 2
Moc-ceph03	10.31.27.204			Ceph Node 3
Moc-mon	10.31.27.206			Monitoring Host
Moc-rgw	10.31.27.223			Rados Gateway

3.1.3 Network Details

Network	IP block	Netmask	Gateway	VLAN
Management	10.31.27.0/24	255.255.255.0	10.31.27.1	2443
External	140.247.152.0/24	255.255.255.0	140.247.152.1	2444
Storage	192.168.27.0/24	255.255.255.0	N/A	600
Out of Band	10.31.29.64/26	255.255.255.0	10.31.29.1	2453

3.2 Satellite 6 Implementation

Satellite was implemented on a single virtual machine. It provides both the content management as well as the installation automation.

3.2.1 Host Preparation

We manually provisioned a RHEL 7 OS instance and then made the following firewall configuration changes:

```
firewall-cmd --add-service=http
firewall-cmd --add-service=http --permanent
firewall-cmd --add-service=https
firewall-cmd --add-service=https --permanent
firewall-cmd --add-port=5671/tcp
firewall-cmd --add-port=5671/tcp --permanent
firewall-cmd --add-port=8140/tcp
firewall-cmd --add-port=8140/tcp  --permanent
firewall-cmd --add-port=9090/tcp
firewall-cmd --add-port=9090/tcp -permanent
```

Now register your host to get the correct packages:

```
subscription-manager subscribe --pool=Red_Hat_Satellite_Pool_Id
subscription-manager repos --disable "*"
subscription-manager repos --enable rhel-6-server-rpms \
--enable rhel-server-rhscl-6-rpms \
--enable rhel-6-server-satellite-6.0-rpms
```

3.2.2 Satellite Installation

Install Katello and run the installation with custom arguments to suit the MOC environment:

```
yum -y install katello
```

```
katello-installer --capsule-dhcp true \
                  --capsule-dhcp-gateway 10.31.27.1 \
                  --capsule-dhcp-interface eth0 \
                  --capsule-dhcp-nameservers 10.31.27.201 \
                  --capsule-dhcp-range "10.31.27.240 10.31.27.250" \
                  --capsule-dns true --capsule-dns-forwarders 10.242.67.11 \
                  --capsule-dns-interface eth0 \
                  --capsule-dns-reverse "27.31.10.in-addr.arpa" \
                  --capsule-dns-zone "moc.rc.fas.harvard.edu" \
                  --capsule-tftp true \
                  --capsule-tftp-servername 10.31.27.201
<...snip..>
Installing             Done                                         [100%]
[.............]
  Success!
  * Katello is running at https://moc-sat.moc.rc.fas.harvard.edu
      Initial credentials are admin / k2nnNFiF7FqPqBSF
  * Capsule is running at https://moc-sat.moc.rc.fas.harvard.edu:9090
  * To install additional capsule on separate machine continue by running:"

      capsule-certs-generate --capsule-fqdn "$CAPSULE" --certs-tar
"~/$CAPSULE-certs.tar"

  The full log is at /var/log/katello-installer/katello-installer.log
root@moc-sat:~[root@moc-sat ~]# exit
```

3.2.3 Setup Credentials for the hammer CLI

Setting up Satellite via the command line is more efficient then the GUI, and can be easily repeated or scripted. First, we need to create a credential file to authenticate the hammer client against the Satellite instance:

```
# cat <<EOF >/etc/hammer/cli_config.yml
> :foreman:
>   :username: 'admin'
>   :password: 'k2nnNFiF7FqPqBSF'
>   :request_timeout: -1
>
> $(cat /etc/hammer/cli_config.yml)
> EOF
```

3.2.4 Populate Satellite with Content

Before importing any Satellite content, we first must create and import a Satellite manifest, generated from the Red Hat Customer Portal. More details on how to create a manifest can be found here: https://access.redhat.com/articles/229083

With the manifest created, we upload it to Satellite:

```
hammer subscription upload --file manifest.zip --organization "Default_Organization"
```

Now we can enable all the Red Hat repositories we need for OpenStack:

```
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat Enterprise Linux Server" \
                             --name "Red Hat Enterprise Linux 7 Server (Kickstart)" \
                             --releasever "7Server" --basearch "x86_64"
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat Enterprise Linux Server" \
                             --name "Red Hat Enterprise Linux 7 Server (RPMs)" \
                             --releasever "7Server" \
                             --basearch "x86_64"
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat Enterprise Linux Server" \
                             --name "Red Hat Enterprise Linux 7 Server - RH Common
(RPMs)" \
                             --releasever "7Server" \
                             --basearch "x86_64"
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat Enterprise Linux Server" \
                             --name "Red Hat Enterprise Linux 7 Server - Optional
(RPMs)" \
                             --releasever "7Server" \
                             --basearch "x86_64"
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat OpenStack" \
                             --name "Red Hat OpenStack 5.0 for RHEL 7 (RPMs)" \
                             --releasever "7Server" \
                             --basearch "x86_64"
hammer repository-set enable --organization "Default_Organization" \
                             --product "Red Hat Software Collections for RHEL Server"
\
                             --name "Red Hat Software Collections RPMs for Red Hat
Enterprise Linux 7 Server" \
                             --releasever "7Server" \
                                         --basearch "x86_64"
```

And do an initial sync of those repositories:

```
for repoid in $(hammer repository list --organization "Default_Organization" | grep
yum | awk '{print $1}'); do

        hammer repository synchronize --id $repoid  --organization
"Default_Organization";

done
```

3.2.5 Configure Environment

First up, we define a subnet for the provisioned hosts in Satellite:

```
hammer subnet create --dhcp-id 1 --dns-id 1 --dns-primary 10.31.27.201 \

                     --domain-ids 1 --from "10.31.27.240" --to "10.31.27.250" \

                     --gateway 10.31.27.1 --mask "255.255.255.0" \

                     --name "moc" --network "10.31.27.0" --tftp-id 1
```

Now we associate all the Satellite infrastructure (dhcp, networks, etc) to our organization and location:

```
hammer organization add-smart-proxy --name "Default_Organization" --smart-proxy "moc-
sat.moc.rc.fas.harvard.edu"

hammer organization add-domain --name "Default_Organization" --domain
"moc.rc.fas.harvard.edu"

hammer organization add-environment --name "Default_Organization" --environment
"production"

hammer organization add-subnet --name "Default_Organization" --subnet "moc"


hammer location add-smart-proxy --name "Default_Location" --smart-proxy "moc-
sat.moc.rc.fas.harvard.edu"

hammer location add-domain --name "Default_Location" --domain
"moc.rc.fas.harvard.edu"

hammer location add-subnet --name "Default_Location" --subnet "moc"

hammer location add-organization --name "Default_Location" --organization
"Default_Organization"

hammer location add-environment --name "Default_Location" --environment "production"

hammer location add-medium --name "Default_Location" --medium
"Default_Organization/Library/Red_Hat_7_Server_Kickstart_x86_64_7Server"
```

3.2.6 Customize the RHEL 7 OS object

With all the software and provisioning services in place, the Red Hat Enterprise Linux 7 OS image can be configured with the proper templates and subscriptions.

```
hammer template add-operatingsystem --name "Satellite Kickstart Default"
--operatingsystem-id 1
```

```
hammer template add-operatingsystem --name "Kickstart default PXELinux"
--operatingsystem-id 1

hammer os add-ptable --id 1  --ptable "Kickstart default"

hammer os set-default-template --id 1 --config-template-id 33

hammer os set-default-template --id 1 --config-template-id 14
```

Activation keys in Satellite allow a system to be registered and automatically subscribed to software and configuration channels. Software is made visible to a system through a mechanism known as a 'content view.' A content view and activation key were created for the OSP systems:

```
hammer content-view create --name "rhel7-osp" --description "OSP repos for RHEL7"
--organization "Default_Organization"

hammer content-view update --name "rhel7-osp" --repository-ids 2,3,4,5,6
--organization "Default_Organization"

hammer content-view publish --name "rhel7-osp" --organization "Default_Organization"

hammer activation-key create --name "rhel-osp-key" --content-view "rhel7-osp"
--lifecycle-environment Library --organization "Default_Organization"

hammer activation-key add-subscription --id 1 --subscription-id
403891bd14cef6f26149cd0c52a30184

hammer global-parameter set --name "kt_org" --value "Default_Organization"

hammer global-parameter set --name "kt_activation_keys" --value "rhel-osp-key"
```

For now, there is no hammer command to manipulate product content associated with activation keys. This step must be done in the GUI. Browse to "Content -> Activation Keys -> rhel-osp-key" and click on "Product Content". Find the entry for "Red Hat OpenStack 5.0 for RHEL 7 (RPMs)" and click "No (Default)", select "Override to Yes" and click "Save."

3.2.7 Add 3rd Party Repositories

For the MOC install, additional packages were required which are not provided by standard Red Hat repositories. These packages were imported as a '3rd party repository' in to Satellite.

```
hammer product list  --organization Default_Organization

hammer product create --organization Default_Organization --label sensu --name Sensu

hammer repository create --organization Default_Organization --content-type yum
--label sensu-x86_64-6 --name "Sensu RHEL 6 x8org/yum/el/6/x86_64su --url
http://repos.sensuapp.

hammer repository synchronize --name "Sensu RHEL 6 x86_64" --product Sensu
--organization Default_Organization

wget https://dl.fedoraproject.org/pub/epel/RPM-GPG-KEY-EPEL-6

wget https://dl.fedoraproject.org/pub/epel/RPM-GPG-KEY-EPEL-7

hammer gpg create --key RPM-GPG-KEY-EPEL-6 --name RPM-GPG-KEY-EPEL-6 --organization
Default_Organization
```

```
hammer gpg create --key RPM-GPG-KEY-EPEL-7 --name RPM-GPG-KEY-EPEL-7 --organization
Default_Organization

hammer repository create --organization Default_Organization --content-type yum
--label epel-x86_64-6 --name "EPEL 6 x86_64" -org/pub/epel/6/x86_64 --gpg-key RPM-
GPG-KEY-EPEL-6

hammer product create --organization Default_Organization --label epel --name EPEL

hammer repository create --organization Default_Organization --content-type yum
--label epel-x86_64-6 --name "EPEL 6 x86_64" -org/pub/epel/6/x86_64 --gpg-key RPM-
GPG-KEY-EPEL-6

hammer repository create --organization Default_Organization --content-type yum
--label epel-x86_64-7 --name "EPEL 7 x86_64" -org/pub/epel/7/x86_64 --gpg-key RPM-
GPG-KEY-EPEL-7

hammer repository synchronize --name "EPEL 6 x86_64" --product EPEL --organization
Default_Organization

wget mirror.seas.harvard.edu/epel/6/x86_64/repodata/repomd.xml

wget mirror.seas.harvard.edu/epel/6/x86_64/repodata/repomd.xml  hammer repository
synchronize --name "EPEL 6 x86_64" --product EPEL --organization Default_Organization

hammer repository synchronize --name "EPEL 6 x76_64" --product EPEL --organization
Default_Organization

hammer repository synchronize --name "EPEL 7 x86_64" --product EPEL --organization
Default_Organization

hammer product create --organization Default_Organization --label elasticsearch
--name elasticsearch

wget http://packages.elasticsearch.org/GPG-KEY-elasticsearch

hammer gpg create --key GPG-KEY-elasticsearch --name GPG-KEY-elasticsearch
--organization Default_Organization

hammer repository create --organization Default_Organization --name "Elasticsearch
1.4" --product elasticsearch --url
http://packages.elasticsearch.org/elasticsearch/1.4/centos

hammer repository synchronize --name "Elasticsearch 1.4" --product elasticsearch
--organization Default_Organization

hammer product create --organization Default_Organization --label logstash --name
logstash

hammer repository create --organization Default_4 --name "Logstash 1.4" --product
logstash --url http://packages.elasticsearch.org/logstash/1.4/centos

hammer repository synchronize --name "Logstash 1.4" --product logstash --organization
Default_Organization

hammer repository create --organization Default_Organization --content-type yum
--label logstashforwarder --name "Logstash Forwarder" --product logstash --url
http://packages.elasticsearch.org/logstashforwarder/centos

hammer repository synchronize --name "Logstash Forwarder" --product logstash
--organization Default_Organization
```

3.3 Ceph Implementation

Ceph was implemented on 3 storage nodes hosting both object storage devices as well as monitors. Due to hardware constraints Ceph was implemented on virtual machines. These virtual machines had one OS disk provided from the hosts mirrored pair. In addition, each Ceph instance had a total of 4 disks passed through. Traditionally you would map 1 journal to 5 OSDs. However, in this case we will keep the journal on disk as we do not have a significant quantity of disks. In addition to the Ceph deployment, one Calamari management node was deployed.

3.3.1 Ceph Prerequisites

Ensure NTP is installed / enabled on all Ceph nodes

```
yum -y install ntp ntpdate

systemctl enable ntpd ; systemctl start ntpd
```

On the Calamari node, ensure the firewall enables port 80 for access. On RHEL 7 (firewalld) this is accomplished with the following command:

```
firewall-cmd --zone=public --add-port=80/tcp

firewall-cmd --zone=public --add-port=80/tcp --permanent

firewall-cmd --zone=public --add-port=443/tcp

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

# Enable access from salt minions

firewall-cmd --zone=public --add-port=4505/tcp

firewall-cmd --zone=public --add-port=4505/tcp --permanent

firewall-cmd --zone=public --add-port=4506/tcp

firewall-cmd --zone=public --add-port=4506/tcp --permanent
```

Note that each Ceph node also needs ports opened. Execute this on each Ceph node:

```
# MON

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

firewall-cmd --zone=public --add-port=6789/tcp

# OSD/MDS

firewall-cmd --zone=public --add-port=6800-7100/tcp --permanent

firewall-cmd --zone=public --add-port=6800-7100/tcp
```

Disable SELinux on all nodes. It is not yet ready to be used with Ceph.

```
setenforce 0

# Permanently disable it

vi /etc/selinux/config

# Set to Permissive or Disabled
```

The servers were registered to the Satellite server and subscribed to the relevant software channels. Software was updated to the latest versions available, and servers were rebooted. On each server:

```
yum -y --nogpgcheck install http://satelliteX.example.com/pub/katello-ca-consumer-latest.noarch.rpm

subscription-manager clean

subscription-manager register --org 'Default_Organization'--activationkey rhel-osp-key

yum install katello-agent

yum -y update

reboot
```

Setup the Inktank Ceph Enterprise (ICE) repository on the Calamari server

```
mkdir /root/ice

scp ICE-1.2-rhel7.tar.gz calamari:/root/ice

cd /root/ice

tar xzvf ICE-1.2-rhel7.tar.gz

python ice_setup.py


--> provide the FQDN for this host: [moc-calamari.moc.rc.fas.harvard.edu]

--> what protocol would this host use (http or https)? [http] http
```

Once the setup completed you will have two new sub-directories. /opt/ICE and /opt/calamari. In addition there will be a cephdeploy.conf in ~/ice.

3.3.2 Calamari Deployment

Initialize Calamari as follows. This will initialize the database and prompt for creating an admin user, password, and email address.

```
calamari-ctl initialize


Username (leave blank to use 'root'):

Email address: root@moc-calamari.moc.rc.fas.harvard.edu

Password: ********
```

At this point you should be able to access Calamari via your web browser!

3.3.3 Ceph Deployment and Configuration

From the calamari server deploy your Ceph nodes and connect them to Calamari

```
ceph-deploy install moc-ceph01 moc-ceph02 moc-ceph03
```

```
   # Note this will SSH into each host.  You need to enter the root password
ceph-deploy calamari connect moc-ceph01 moc-ceph02 moc-ceph03
```

NOTE: We probably shouldn't add the nodes in the GUI until the Ceph cluster is running.

Once done with the connect, log into the Calamari GUI and under the Cluster tab approve the pending hosts.

NOTE: Before you approve the hosts, /var/log/salt/minions will give the following message every 10 seconds. This is expected as the node is not approved yet:

```
        2014-11-11 16:40:40,412 [salt.crypt       ][ERROR   ] The Salt Master has cached the public key  for this node, this salt minion will wait for 10 seconds before attempting to re-authenticate
```

Now we can configure Ceph MONs and OSDs.

```
#Configure new cluster (Execute from Calamari host)
mkdir /etc/ceph
cd /etc/ceph
ceph-deploy new moc-ceph01 moc-ceph02 moc-ceph03


#Create monitors
ceph-deploy mon create-initial
# At this point ceph health will show the cluster in Error.  This is expected


#Gather Keys
ceph-deploy gatherkeys moc-ceph01


#Create OSDs, (NOTE: 5 disks per journal is recommended.  We're using local journals as we only have 4 disk)


ceph-deploy osd create moc-ceph01:vd{b..e} --zap


ceph-deploy osd create moc-ceph02:vd{b..e} --zap


ceph-deploy osd create moc-ceph03:vd{b..e} --zap


ceph -s
ceph osd stat
ceph osd tree
```

Note – Ceph may show HEALTH_WARN too few pgs per osd. This is fine for now and we do NOT need to change the default 3 pools. When we add OpenStack pools we will get the correct number of placement groups.

Optimal placement groups are documented in http://ceph.com/docs/master/rados/operations/placement-groups/.

We have a total of 12 OSDs today. The documentation recommends 4096 placement groups for 10-50 OSDs. However, that's a wide range. (12 OSDs * 100) / 3 replicas = 400. So we could go as low as 512 placement groups.

3.3.4 Ceph OpenStack Integration

Now that Ceph is setup, we can do the integration with OpenStack. In our case, OpenStack is being deployed from Satellite (Foreman + Puppet). So we need to have the Ceph environment prepared to work with this. Three components need to be setup with Ceph: Nova Compute, Glance API, and Cinder API.

Setup Ceph for OpenStack. Log into one of your storage nodes for this.

```
ssh moc-ceph01
cd /etc/ceph

# (optional) Set the default placement groups aligned with the above
### Note - I'm skipping this.  I'll specify placement groups on pool create
#vi /etc/ceph/ceph.conf      # In the [global] section
#osd pool default pg num = 2048
#osd pool default pgp num = 2048
# Repeat for moc-ceph02, moc-ceph03, AND moc-calamari

# NOW, back on moc-ceph01:
#First create the pools for images and volumes
## NOTE - I selected placement groups of 2048.  Compromised between 512 and 4096 :)
ceph osd pool create images 2048
ceph osd pool create volumes 2048

# Create Authentication Keyrings for Glance and cinder
ceph auth get-or-create client.cinder mon 'allow r' osd 'allow class-read
object_prefix rbd_children, allow rwx pool=volumes, allow rx pool=images' -o
/etc/ceph/client.cinder.key
ceph auth get-or-create client.glance mon 'allow r' osd 'allow class-read
object_prefix rbd_children, allow rwx pool=images' -o /etc/ceph/client.glance.key
# Note - The client.nova is used by Foreman / Puppet modules to generate the libvirt
secret key for nova...
ceph auth get-or-create client.nova mon 'allow r' osd 'allow class-read object_prefix
rbd_children, allow rwx pool=volumes, allow rx pool=images' -o
/etc/ceph/client.nova.key

# Copy these to your Calamari host for safe keeping
scp /etc/ceph/client.*.key moc-calamari:/etc/ceph
```

```
# Create a UUID to share amongst the cluster:
uuidgen
### NOTE: You will need this for your host group config below
## For example: 519a58bd-1980-44e1-9ca7-4f480fa02566

# Add your client keyring detail to /etc/ceph/ceph.conf.  Otherwise your controller
won't be able to authenticate and will give a non-descript error
cat << EOF >> /etc/ceph/ceph.conf
[client.cinder]
keyring = /etc/ceph/client.cinder.key

[client.glance]
keyring = /etc/ceph/client.glance.key

[client.nova]
keyring = /etc/ceph/client.nova.key
EOF

# Distribute /etc/ceph/ceph.conf and your client keys to ALL nodes (both controllers
and compute).  The host groups don't do anything to generate this...
for host in controller compute0{1..9} compute{11..15}
do
  ssh $host 'mkdir -p /etc/ceph'
  scp ceph.conf $host:/etc/ceph
  scp client.*.key $host:/etc/ceph
done
```

3.4 OpenStack Implementation

3.4.1 Preparation

Following from the work of the OpenStack Foreman Installer (OFI), puppet modules and host groups were imported in to Satellite.

```
# stage modules for import
cd /opt/osp/
mkdir modules
mkdir src
```

```
cd src

cp /var/lib/pulp/content/rpm/openstack-puppet-
modules/2014.1/24.el7ost/noarch/f175173084c58277ed464f112c108ddfa3aad117/openstack-
puppet-modules-2014.1-24.el7ost.noarch.rpm .

rpm2cpio openstack-puppet-modules-2014.1-24.el7ost.noarch.rpm | cpio -idmv

rsync -avh --progress /opt/osp/src/usr/share/openstack-foreman-
installer/puppet/modules/* /opt/osp/modules/

wget https://github.com/redhat-openstack/astapor/archive/openstack-foreman-installer-
2.0.32.tar.gz

tar -xzvf openstack-foreman-installer-2.0.32.tar.gz

rsync -avh --progress /opt/osp/src/astapor-openstack-foreman-installer-
2.0.32/puppet/modules/* /opt/osp/modules/

restorecon -Rv /opt/osp/modules/

mkdir /etc/puppet/environments/production

ln -s /opt/osp/modules /etc/puppet/environments/production

# import modules in to Satellite

hammer proxy import-classes --environment production --id 1
```

The OFI RPM provides a seeds file that will populate the Satellite database with the proper host groups and set sane default parameters for the puppet modules. It needs to be modified slightly to work with Satellite.

```
cp /opt/osp/src/astapor-openstack-foreman-installer-2.0.32/bin/bin/seeds.rb
/usr/share/foreman/seeds.d/99-quickstack.rb
```

Modify /usr/share/foreman/seeds.d/99-quickstack.rb. Remove all the lines after the 2 require declarations:

```
require 'facter'

require 'securerandom'
```

Up to but not including the line that starts with:

```
'params = {'
```

Skip down to the block at the end of the file starting with:

```
if ENV["FOREMAN_PROVISIONING"] == "true" then
```

and delete all lines from there to the end of the file.

Finally, the database is reseeded with the OpenStack related data, and the newly created host groups were associated to the organization:

```
sudo -u foreman scl enable ruby193 "cd /usr/share/foreman; export
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/rh/v8314/root/usr/lib64:/opt/rh/ruby193/root/li
b64/; rake db:seed RAILS_ENV=production"

for hostgroup in $(hammer hostgroup list | grep production | awk '{print $1}'); do
hammer organization add-hostgroup --name "Default_Organization" --hostgroup-id
$hostgroup; done

for hostgroup in $(hammer hostgroup list | grep production | awk '{print $1}'); do
hammer location add-hostgroup --name "Default_Location" --hostgroup-id $hostgroup;
done
```

3.4.2 Base OS Provisioing

The OpenStack hosts were first provisioned with Satellite before applying the puppet modules. The steps to do this are reproduced below:

1. In the Satellite GUI, browse to "Hosts -> New host".
2. Enter the name for the controller node, choose 'production' for 'Puppet Environment' and select the Satellite server in the 3 drop down lists under 'Capsule Settings.'
3. Click 'Network' and select Satellite domain in the 'Domain' drop down list. Enter the MAC address of the boot interface of the server, select the Satellite subnet, and allow the GUI to auto-suggest an IP.
4. Under 'Operating System' select the proper Architecture, Operating system, Media and Partition table from the respective drop downs. Enter a root password, click 'Resolve' next 'provisioning templates', and finally hit submit. .
5. Boot the server and allow Satellite to provision an operating system. Login and configure the primary and secondary network interfaces to use statically assigned IPs, as documented.
6. Repeat for the network node and any compute nodes desired.

3.4.3 OpenStack Base Installation

To deploy OpenStack with Satellite, hosts are simply assigned to the relevant host group and the configuration will be applied by either a manual puppet run, or during the next poll to the Satellite server.

Before doing that, confirm the host group parameters are accurate.

NOTE: We created a custom puppet manifest for this engagement for the controller. This was called MOC controller and is detailed in the appendix. We used this to enable a converged controller and networker node in a non-HA configuration. The 'Compute (Neutron)' host group was used for the compute nodes.

In the Satellite GUI, Browse to "Configure -> Host groups -> Controller (Neutron) -> Parameters" and override the variables to suit the environment. Repeat for the "Compute (Neutron)" and "Neutron Networker." This step may not be necessary if you fully modified the seeds file appropriately before running the rake command earlier.

Once all parameters are confirmed, OpenStack can be deployed:

1. Browse to "Host -> All hosts" and select the controller node. Click "edit" and under "Host Group" select "Controller (Neutron)" and hit "submit."
2. Login in to the controller and run 'puppet agent -t' or wait until next scheduled puppet run.
3. Repeat these steps for the networker node, followed by any controller nodes.

3.4.4 OpenStack Ceph Manual Configuration

Puppet handles most of configuring OpenStack to talk with Ceph. However, it is still missing the Nova ephemeral configuration. This needs to be done manually for each compute node as follows:

```
cat << EOF >> /tmp/novasecret.xml
```

```
<secret ephemeral='no' private='no'>
  <uuid>519a58bd-1980-44e1-9ca7-4f480fa02566</uuid>
  <usage type='ceph'>
    <name>client.cinder secret</name>
  </usage>
</secret>
EOF

cat /etc/ceph/client.cinder.key
# Grab the value for your base64 command

for server in compute0{1..9} compute{10..15}
do
   scp /tmp/novasecret.xml $server:/tmp
   ssh $server 'virsh secret-define --file /tmp/novasecret.xml'
   for server in compute0{3..9} compute{11..15}; do ssh $server 'virsh secret-set-value --secret 519a58bd-1980-44e1-9ca7-4f480fa02566 --base64 AQAJfGVU4P0OLBAAse4ozRODg18OlnuJ9LBrTQ=='
done

for server in compute0{1..9} compute{10..15}
do
  ssh $server 'systemctl restart openstack-nova-compute'
done
```

3.4.5 OpenStack Base Validation

```
ssh controller
cd /root
. keystonerc_admin
# Upload test image
wget http://download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
### NOTE - For Ceph you need to use a raw format!  Convert the image to RAW...
qemu-img convert -f qcow2 -O raw cirros-0.3.3-x86_64-disk.img cirros-0.3.3-x86_64-disk.raw
glance image-create --name "Cirros-0.3.3-x86_64" --disk-format qcow2 --container-format bare --is-public true --file cirros-0.3.3-x86_64-disk.raw
# Enable ping/SSH in default security group
neutron security-group-rule-create --protocol icmp --direction ingress default
neutron security-group-rule-create --protocol tcp --port-range-min 22 --port-range-
max 22 --direction ingress default


# Setup SSH keypair for admin

nova keypair-add adminkey > /root/adminkey.pem

chmod 600 /root/adminkey.pem

neutron net-create pubnet1 --shared --provider:physical_network physext1
--provider:network_type flat --router:external=True

neutron subnet-create --name pubsubnet --dns-nameserver 8.8.8.8 --allocation-pool
start=140.247.152.2,end=140.247.152.200 --disable-dhcp pubnet1 140.247.152.0/24

### NOTE - 205 is the controller.  Probably better to put this later in the range...

PUBNETID=$(neutron net-list | grep pubnet1 | awk '{print $2}')



#Creating Internal L2 private networks

neutron net-create privnet1

neutron subnet-create --name privsubnet1 privnet1 10.0.0.0/24

neutron net-create privnet2

neutron subnet-create --name privsubnet2 privnet2 10.0.1.0/24


# Capture Net/Subnet UUIDs

PRIVNET1ID=$(neutron net-list | grep privnet1 | awk '{print $2}')

PRIVNET2ID=$(neutron net-list | grep privnet2 | awk '{print $2}')

PRIVSUBNET1ID=$(neutron subnet-list | grep privsubnet1 | awk '{print $2}')

PRIVSUBNET2ID=$(neutron subnet-list | grep privsubnet2 | awk '{print $2}')


# Create router for L3 to connect each network

neutron router-create router1

neutron router-interface-add router1 $PRIVSUBNET1ID

neutron router-interface-add router1 $PRIVSUBNET2ID


# Set the gateway for your external (physical) network

neutron router-gateway-set router1 $PUBNETID


# Boot a test image

nova boot testserver --flavor m1.tiny --image Cirros-0.3.3-x86_64 --key-name adminkey
--security-groups default --nic net-id=$PRIVNET1ID


# Assign a floating IP and test connectivity
```

```
FLOATINGOUTPUT=$(neutron floatingip-create pubnet1)
FLOATINGIP=$(echo $FLOATINGOUTPUT | awk -F "|" '{print $9}' | sed 's/ //g')
nova add-floating-ip testserver $FLOATINGIP
# Validate connectivity (and that the guest can ping externally
ssh -i /root/adminkey.pem cirros@$FLOATINGIP 'ping -c 5 8.8.8.8'
```

Note that outside of the controller you will not be able to connect to the floating IP. This is because we do not yet have the external network firewall open. In other words, the IP's we plan to use for external connectivity are currently blocked by physical firewall external to the OpenStack deployment.

4 ISSUES

4.1 Ceph Redeploy

When we rebuilt the environment we ended up corrupting the Ceph build. Rather than recover it was felt it would be quicker to reinstall. However, we ran into an issue where we couldn't get the OSD's to redeploy. It turned out there was an auth error on the bootstrap-osd which was hidden. To clear this issue I did the following:

```
# On moc-ceph01-s, 02-s, and 03-s:
rm -rf /etc/ceph
[root@moc-ceph02-s ~]# rm -rf /var/lib/ceph/*
[root@moc-ceph02-s ~]# sgdisk /dev/vdb --zap-all
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
[root@moc-ceph02-s ~]# sgdisk /dev/vdc --zap-all
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
[root@moc-ceph02-s ~]# sgdisk /dev/vdd --zap-all
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
[root@moc-ceph02-s ~]# sgdisk /dev/vde --zap-all
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
mkdir -p /var/lib/ceph/mon
mkdir -p /var/lib/ceph/tmp
# Starting with ceph0-3 on node 1, ceph-4-6 on node 2, and ceph-7-9 on node 3
mkdir -p /var/lib/ceph/osd/ceph-4
mkdir -p /var/lib/ceph/osd/ceph-5
mkdir -p /var/lib/ceph/osd/ceph-6
mkdir /etc/ceph

# Then on moc-calamari
cd /etc/ceph
rm -rf *
ceph-deploy new moc-ceph01-s moc-ceph02-s moc-ceph03-s
ceph-deploy mon create-initial
ceph-deploy gatherkeys moc-ceph01-s
ceph-deploy osd create moc-ceph01-s:vd{b..e} --zap
ceph-deploy osd create moc-ceph02-s:vd{b..e} --zap
```

```
ceph-deploy osd create moc-ceph03-s:vd{b..e} -zap




### NOT RELATED TO ABOVE:

ceph auth list          # lists all the authorization
```

4.2 VLAN tenant networks

We spent a large amount of time trying to get a single interface to be used as a trunk of VLANs for tenants as well as another standalone VLAN for external network access. After much troubleshooting as well as discussing internally with a number of people, it was decided that this configuration will not work. Since we do not have an alternative with VLAN based on our switch configuration, we decided to redeploy with VxLAN. VxLAN allows us to split the VLAN for external network and also a VLAN for tenant networking (with VxLAN layed over the top for tenant segregation).

4.3 SSL Implementation

We found that currently the SSL configuration of RabbitMQ does not complete successfully from the Foreman/Puppet install. To temporarily work around this we have disabled SSL. However, we need to address this moving forward as SSL will be important in a public deployment. Note that HAProxy can be used to offload SSL allowing the underlying cloud deployment to remain non-SSL.

5 NEXT STEPS

In general we have a much greater scope than what we can deliver in the time we had available. I also think that will continue. We came into this not so much thinking about overall requirements, but about the single deployment at hand. In hind site I feel that was a mistake. We really need to focus on how we're going to deliver at scale and provide operational support ongoing. We need to focus on the these tasks:

- Production processes
  - Patch management
  - Monitoring
  - Ongoing configuration management
  - Backups
  - Upgrades
- Repeatable installation processes
  - Process to test/dev prior to promotion
- Training/Mentoring of support staff
- High Availability
- Security for a public cloud
- Further Technical Validation
  - Sahara
  - LBaas, FWaaS, VPNaaS

I feel it best to begin scoping requirements, growth, and timeline so we are able to prioritize these needs and find a way to deliver against them.

6 APPENDIXES

6.1 Host Group Parameters

The host groups require parameter configuration to tailor the installation to the specific environment. Below are the host group parameters overridden for this install. Note that the passwords have been replaced with ******* for security reasons.

6.1.1 MOC Controller

Host group parameters

quickstack::neutron:	controller_admin_hos	10.31.27.207
quickstack::neutron:	amqp_host	10.31.27.207
quickstack::neutron:	admin_password	********
quickstack::neutron:	amqp_nssdb_password	********
quickstack::neutron:	amqp_password	********
quickstack::neutron:	cinder_backend_rbd	true
quickstack::neutron:	ceilometer_user_pass	********
quickstack::neutron:	ovs_tunnel_iface	em1
quickstack::neutron:	cinder_user_passworc	********
quickstack::neutron:	cinder_rbd_secret_uu	519a58bd-1980-44e1-9ca7-4f480fa02566
quickstack::neutron:	glance_backend	rbd
quickstack::neutron:	sahara	True
quickstack::neutron:	ml2_tenant_network_t	["vxlan","flat"]
quickstack::neutron:	ml2_type_drivers	["vxlan","flat"]
quickstack::neutron:	ml2_vni_ranges	["10:10000"]
quickstack::neutron:	cinder_db_password	********
quickstack::neutron:	sahara_user_passworc	********
quickstack::neutron:	ovs_bridge_uplinks	["br-ex:em1.2444"]
quickstack::neutron:	neutron_user_passwor	********

quickstack::neutron:	nova_db_password	********
quickstack::neutron:	keystonerc	true
quickstack::neutron:	mysql_host	10.31.27.207
quickstack::neutron:	controller_pub_host	140.247.152.207
quickstack::neutron:	controller_priv_host	10.31.27.207
quickstack::neutron:	glance_db_password	********
quickstack::neutron:	glance_user_passworc	********
quickstack::neutron:	heat_cloudwatch	true
quickstack::neutron:	heat_db_password	********
quickstack::neutron:	heat_user_password	********
quickstack::neutron:	keystone_db_passworc	********
quickstack::neutron:	mysql_root_password	********
quickstack::neutron:	sahara_use_neutron	True
quickstack::neutron:	sahara_db_password	********
quickstack::neutron:	enable_tunneling	True
quickstack::neutron:	external_network_bri	br-ex
quickstack::neutron:	ovs_bridge_mappings	["physext1:br-ex"]
quickstack::neutron:	neutron_db_password	********
quickstack::neutron:	nova_user_password	********
quickstack::neutron:	glance_rbd_store_use	glance
quickstack::neutron:	cinder_rbd_user	cinder
quickstack::neutron:	ssl	false

6.1.2 Compute (Neutron)

Host group parameters

quickstack::neutron:	rbd_secret_uuid	519a58bd-1980-44e1-9ca7-4f480fa02566
quickstack::neutron:	admin_password	********
quickstack::neutron:	amqp_password	********
quickstack::neutron:	auth_host	10.31.27.207
quickstack::neutron:	ceilometer_user_pass	********
quickstack::neutron:	glance_host	10.31.27.207
quickstack::neutron:	nova_host	10.31.27.207
quickstack::neutron:	nova_user_password	********
quickstack::neutron:	amqp_host	10.31.27.207
quickstack::neutron:	neutron_db_password	********
quickstack::neutron:	ssl	false
quickstack::neutron:	neutron_user_passwor	********
quickstack::neutron:	tunnel_id_ranges	10:10000
quickstack::neutron:	mysql_host	10.31.27.207
quickstack::neutron:	nova_db_password	********
quickstack::neutron:	neutron_host	10.31.27.207
quickstack::neutron:	cinder_backend_rbd	True
quickstack::neutron:	enable_tunneling	True
quickstack::neutron:	private_iface	em1
quickstack::neutron:	ovs_tunnel_iface	em1
quickstack::neutron:	rbd_user	cinder

6.2 Puppet Modifications Used

The below are a list of puppet modifications done to enable this deployment.

6.2.1 Consolidated Quickstack controller / networker

```
cat /usr/share/openstack-foreman-
installer/puppet/modules/quickstack/manifests/neutron/moc-all.pp

class quickstack::neutron::moc-all (

  $admin_email                     = $quickstack::params::admin_email,

  $admin_password                  = $quickstack::params::admin_password,

  $auth_host                       = 'localhost',

  $auth_tenant                     = 'services',

  $auth_user                       = 'neutron',

  $ceilometer_metering_secret      = $quickstack::params::ceilometer_metering_secret,

  $ceilometer_user_password        = $quickstack::params::ceilometer_user_password,

  $cinder_backend_eqlx             = $quickstack::params::cinder_backend_eqlx,

  $cinder_backend_eqlx_name        = $quickstack::params::cinder_backend_eqlx_name,

  $cinder_backend_gluster          = $quickstack::params::cinder_backend_gluster,

  $cinder_backend_gluster_name     = $quickstack::params::cinder_backend_gluster_name,

  $cinder_backend_iscsi            = $quickstack::params::cinder_backend_iscsi,

  $cinder_backend_iscsi_name       = $quickstack::params::cinder_backend_iscsi_name,

  $cinder_backend_nfs              = $quickstack::params::cinder_backend_nfs,

  $cinder_backend_nfs_name         = $quickstack::params::cinder_backend_nfs_name,

  $cinder_backend_rbd              = $quickstack::params::cinder_backend_rbd,

  $cinder_backend_rbd_name         = $quickstack::params::cinder_backend_rbd_name,

  $cinder_db_password              = $quickstack::params::cinder_db_password,

  $cinder_gluster_shares           = $quickstack::params::cinder_gluster_shares,

  $cinder_nfs_shares               = $quickstack::params::cinder_nfs_shares,

  $cinder_nfs_mount_options        = $quickstack::params::cinder_nfs_mount_options,

  $cinder_san_ip                   = $quickstack::params::cinder_san_ip,

  $cinder_san_login                = $quickstack::params::cinder_san_login,

  $cinder_san_password             = $quickstack::params::cinder_san_password,

  $cinder_san_thin_provision       = $quickstack::params::cinder_san_thin_provision,

  $cinder_eqlx_group_name          = $quickstack::params::cinder_eqlx_group_name,

  $cinder_eqlx_pool                = $quickstack::params::cinder_eqlx_pool,
  $cinder_eqlx_use_chap            = $quickstack::params::cinder_eqlx_use_chap,
  $cinder_eqlx_chap_login          = $quickstack::params::cinder_eqlx_chap_login,
  $cinder_eqlx_chap_password       = $quickstack::params::cinder_eqlx_chap_password,
  $cinder_rbd_pool                 = $quickstack::params::cinder_rbd_pool,
  $cinder_rbd_ceph_conf            = $quickstack::params::cinder_rbd_ceph_conf,
  $cinder_rbd_flatten_volume_from_snapshot
                                   =
$quickstack::params::cinder_rbd_flatten_volume_from_snapshot,
  $cinder_rbd_max_clone_depth      = $quickstack::params::cinder_rbd_max_clone_depth,
  $cinder_rbd_user                 = $quickstack::params::cinder_rbd_user,
  $cinder_rbd_secret_uuid          = $quickstack::params::cinder_rbd_secret_uuid,
  $cinder_user_password            = $quickstack::params::cinder_user_password,
  $cisco_nexus_plugin              = $quickstack::params::cisco_nexus_plugin,
  $cisco_vswitch_plugin            = $quickstack::params::cisco_vswitch_plugin,
  $controller_admin_host           = $quickstack::params::controller_admin_host,
  $controller_priv_host            = $quickstack::params::controller_priv_host,
  $controller_pub_host             = $quickstack::params::controller_pub_host,
  $external_network_bridge         = '',
  $fixed_network_range             = $quickstack::params::fixed_network_range,
  $fwaas                           = $quickstack::params::fwaas,
  $glance_db_password              = $quickstack::params::glance_db_password,
  $glance_user_password            = $quickstack::params::glance_user_password,
  $glance_backend                  = $quickstack::params::glance_backend,
  $glance_rbd_store_user           = $quickstack::params::glance_rbd_store_user,
  $glance_rbd_store_pool           = $quickstack::params::glance_rbd_store_pool,
  $heat_auth_encrypt_key,
  $heat_cfn                        = $quickstack::params::heat_cfn,
  $heat_cloudwatch                 = $quickstack::params::heat_cloudwatch,
  $heat_db_password                = $quickstack::params::heat_db_password,
  $heat_user_password              = $quickstack::params::heat_user_password,
  $horizon_secret_key              = $quickstack::params::horizon_secret_key,
  $keystone_admin_token            = $quickstack::params::keystone_admin_token,
  $keystone_db_password            = $quickstack::params::keystone_db_password,
  $keystonerc                      = false,
  $lbaas                  = $quickstack::params::lbaas,
  $neutron                         = $quickstack::params::neutron,
  $neutron_metadata_proxy_secret =
$quickstack::params::neutron_metadata_proxy_secret,
  $neutron_metering_agent      = $quickstack::params::neutron_metering_agent,
  $mysql_host                     = $quickstack::params::mysql_host,
  $mysql_root_password            = $quickstack::params::mysql_root_password,
  $neutron_core_plugin            = 'neutron.plugins.ml2.plugin.Ml2Plugin',
  $neutron_db_password            = $quickstack::params::neutron_db_password,
  $neutron_user_password          = $quickstack::params::neutron_user_password,
  $nexus_config                   = $quickstack::params::nexus_config,
  $nexus_credentials              = $quickstack::params::nexus_credentials,
  $nova_db_password               = $quickstack::params::nova_db_password,
  $nova_user_password             = $quickstack::params::nova_user_password,
  $nova_default_floating_pool     = $quickstack::params::nova_default_floating_pool,
  $ovs_vlan_ranges                = $quickstack::params::ovs_vlan_ranges,
  $ovs_tunnel_iface               = 'eth1',
  $ovs_tunnel_network             = '',
  $ovs_l2_population              = 'True',
  $ovs_bridge_mappings            = $quickstack::params::ovs_bridge_mappings,
  $ovs_bridge_uplinks             = $quickstack::params::ovs_bridge_uplinks,
  $ovs_vxlan_udp_port             = $quickstack::params::ovs_vxlan_udp_port,
  $ovs_tunnel_types               = $quickstack::params::ovs_tunnel_types,
  $provider_vlan_auto_create      = $quickstack::params::provider_vlan_auto_create,
  $provider_vlan_auto_trunk       = $quickstack::params::provider_vlan_auto_trunk,
  $enable_tunneling               = $quickstack::params::enable_tunneling,
  $tunnel_id_ranges               = '1:1000',
  $ml2_type_drivers               = ['vxlan', 'flat','vlan','gre','local'],
  $ml2_tenant_network_types       = ['vxlan', 'flat','vlan','gre'],
  $ml2_mechanism_drivers          = ['openvswitch','l2population'],
  $ml2_flat_networks              = ['*'],
  $ml2_network_vlan_ranges        = ['physnet1:1000:2999'],
  $ml2_tunnel_id_ranges           = ['20:100'],
  $ml2_vxlan_group                = '224.0.0.1',
  $ml2_vni_ranges                 = ['10:100'],
  $ml2_security_group             = 'true',
  $ml2_firewall_driver            =
'neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver',
  $amqp_provider                  = $quickstack::params::amqp_provider,
  $amqp_host                      = $quickstack::params::amqp_host,
  $amqp_username                  = $quickstack::params::amqp_username,
  $amqp_password                  = $quickstack::params::amqp_password,
  $amqp_nssdb_password            = $quickstack::params::amqp_nssdb_password,
  $sahara               = $quickstack::params::sahara,
  $sahara_db_password          = $quickstack::params::sahara_db_password,
  $sahara_user_password              = $quickstack::params::sahara_user_password,
  $sahara_use_neutron          = $quickstack::params::sahara_use_neutron,
  $swift_shared_secret            = $quickstack::params::swift_shared_secret,
  $swift_admin_password           = $quickstack::params::swift_admin_password,
  $swift_ringserver_ip            = '192.168.203.1',
  $swift_storage_ips              = ['192.168.203.2', '192.168.203.3',
'192.168.203.4'],
  $swift_storage_device           = 'device1',
  $tenant_network_type            = $quickstack::params::tenant_network_type,
  $verbose                        = $quickstack::params::verbose,
  $vpnaas                         = $quickstack::params::vpnaas,
  $ssl                            = $quickstack::params::ssl,
  $freeipa                        = $quickstack::params::freeipa,
  $mysql_ca                       = $quickstack::params::mysql_ca,
  $mysql_cert                     = $quickstack::params::mysql_cert,
  $mysql_key                      = $quickstack::params::mysql_key,
  $amqp_ca                        = $quickstack::params::amqp_ca,
  $amqp_cert                      = $quickstack::params::amqp_cert,
  $amqp_key                       = $quickstack::params::amqp_key,
  $horizon_ca                     = $quickstack::params::horizon_ca,
  $horizon_cert                   = $quickstack::params::horizon_cert,
  $horizon_key                    = $quickstack::params::horizon_key,

) inherits quickstack::params {

  if str2bool_i("$ssl") {
    $qpid_protocol = 'ssl'
    $amqp_port = '5671'
    $sql_connection = "mysql://neutron:${neutron_db_password}@${mysql_host}/neutron?
ssl_ca=${mysql_ca}"
  } else {
    $qpid_protocol = 'tcp'
    $amqp_port = '5672'
    $sql_connection = "mysql://neutron:${neutron_db_password}@${mysql_host}/neutron"
  }

  class { 'quickstack::controller_common':
    admin_email                      => $admin_email,
    admin_password                   => $admin_password,
    ceilometer_metering_secret       => $ceilometer_metering_secret,
    ceilometer_user_password         => $ceilometer_user_password,
    cinder_backend_eqlx              => $cinder_backend_eqlx,
    cinder_backend_eqlx_name         => $cinder_backend_eqlx_name,
    cinder_backend_gluster           => $cinder_backend_gluster,
    cinder_backend_gluster_name      => $cinder_backend_gluster_name,
    cinder_backend_iscsi             => $cinder_backend_iscsi,
    cinder_backend_iscsi_name        => $cinder_backend_iscsi_name,
    cinder_backend_nfs               => $cinder_backend_nfs,
    cinder_backend_nfs_name          => $cinder_backend_nfs_name,
    cinder_backend_rbd               => $cinder_backend_rbd,
    cinder_backend_rbd_name          => $cinder_backend_rbd_name,
    cinder_db_password               => $cinder_db_password,
    cinder_multiple_backends         => $cinder_multiple_backends,
    cinder_gluster_shares            => $cinder_gluster_shares,
    cinder_nfs_shares                => $cinder_nfs_shares,
    cinder_nfs_mount_options         => $cinder_nfs_mount_options,
    cinder_san_ip                    => $cinder_san_ip,
    cinder_san_login                 => $cinder_san_login,
    cinder_san_password              => $cinder_san_password,
    cinder_san_thin_provision        => $cinder_san_thin_provision,
    cinder_eqlx_group_name           => $cinder_eqlx_group_name,
    cinder_eqlx_pool                 => $cinder_eqlx_pool,
    cinder_eqlx_use_chap             => $cinder_eqlx_use_chap,
    cinder_eqlx_chap_login           => $cinder_eqlx_chap_login,
    cinder_eqlx_chap_password        => $cinder_eqlx_chap_password,
    cinder_rbd_pool                  => $cinder_rbd_pool,
    cinder_rbd_ceph_conf             => $cinder_rbd_ceph_conf,
    cinder_rbd_flatten_volume_from_snapshot
                                     => $cinder_rbd_flatten_volume_from_snapshot,
    cinder_rbd_max_clone_depth       => $cinder_rbd_max_clone_depth,
    cinder_rbd_user                  => $cinder_rbd_user,
    cinder_rbd_secret_uuid           => $cinder_rbd_secret_uuid,
    cinder_user_password             => $cinder_user_password,
    controller_admin_host            => $controller_admin_host,
    controller_priv_host             => $controller_priv_host,
    controller_pub_host             => $controller_pub_host,
    glance_db_password              => $glance_db_password,
    glance_user_password            => $glance_user_password,
    glance_backend                  => $glance_backend,
    glance_rbd_store_user           => $glance_rbd_store_user,
    glance_rbd_store_pool           => $glance_rbd_store_pool,
    heat_auth_encrypt_key           => $heat_auth_encrypt_key,
    heat_cfn                        => $heat_cfn,
    heat_cloudwatch                 => $heat_cloudwatch,
    heat_db_password                => $heat_db_password,
    heat_user_password              => $heat_user_password,
    horizon_secret_key              => $horizon_secret_key,
    keystone_admin_token            => $keystone_admin_token,
    keystone_db_password            => $keystone_db_password,
    keystonerc                      => $keystonerc,
    neutron_metadata_proxy_secret => $neutron_metadata_proxy_secret,
    mysql_host                      => $mysql_host,
    mysql_root_password             => $mysql_root_password,
    neutron                         => true,
    neutron_core_plugin             => $neutron_core_plugin,
    neutron_db_password             => $neutron_db_password,
    neutron_user_password           => $neutron_user_password,
    nova_db_password                => $nova_db_password,
    nova_user_password              => $nova_user_password,
    nova_default_floating_pool      => $nova_default_floating_pool,
    amqp_host                       => $amqp_host,
    amqp_username                   => $amqp_username,
    amqp_password                   => $amqp_password,
    sahara                          => $sahara,
    sahara_db_password              => $sahara_db_password,
    sahara_user_password            => $sahara_user_password,
    sahara_use_neutron              => $sahara_use_neutron,
    swift_shared_secret             => $swift_shared_secret,
    swift_admin_password            => $swift_admin_password,
    swift_ringserver_ip             => $swift_ringserver_ip,
    swift_storage_ips               => $swift_storage_ips,
    swift_storage_device            => $swift_storage_device,
    verbose                         => $verbose,
    ssl                              => $ssl,
    freeipa                          => $freeipa,
    mysql_ca                         => $mysql_ca,
    mysql_cert                       => $mysql_cert,
    mysql_key                        => $mysql_key,
    amqp_ca                          => $amqp_ca,
    amqp_cert                        => $amqp_cert,
    amqp_key                         => $amqp_key,
    horizon_ca                       => $horizon_ca,
    horizon_cert                     => $horizon_cert,
    horizon_key                      => $horizon_key,
    amqp_nssdb_password              => $amqp_nssdb_password,
  }
  ->
  class { '::neutron':
    enabled                 => true,
    verbose                 => $verbose,
    allow_overlapping_ips => true,
    rpc_backend             => amqp_backend('neutron', $amqp_provider),
    qpid_hostname           => $amqp_host,
    qpid_port               => $amqp_port,
    qpid_protocol           => $qpid_protocol,
    qpid_username           => $amqp_username,
    qpid_password           => $amqp_password,
    rabbit_host             => $amqp_host,
    rabbit_port             => $amqp_port,
    rabbit_user             => $amqp_username,
    rabbit_password         => $amqp_password,
    core_plugin             => $neutron_core_plugin
  }
  ->
  class { '::nova::network::neutron':
    neutron_admin_password => $neutron_user_password,
  }
  ->
  class { '::neutron::server::notifications':
    notify_nova_on_port_status_changes => true,
    notify_nova_on_port_data_changes   => true,
    nova_url                          => "http://${controller_priv_host}:8774/v2",
    nova_admin_auth_url               => "http://$
{controller_priv_host}:35357/v2.0",
    nova_admin_username               => "nova",
    nova_admin_password               => "${nova_user_password}",
  }
  ->
  # FIXME: This really should be handled by the neutron-puppet module, which has
  # a review request open right now: https://review.openstack.org/#/c/50162/
  # If and when that is merged (or similar), the below can be removed.
  exec { 'neutron-db-manage upgrade':
    command     => 'neutron-db-manage --config-file /etc/neutron/neutron.conf
--config-file /etc/neutron/plugin.ini upgrade head',
    path        => '/usr/bin',
    user        => 'neutron',
    logoutput   => 'on_failure',
    before      => Service['neutron-server'],
    require     => [Neutron_config['database/connection'],
Neutron_config['DEFAULT/core_plugin']],
  }

  #neutron_config {
  #   'keystone_authtoken/auth_host':         value => $auth_host;
  #   'keystone_authtoken/admin_tenant_name': value => 'services';
  #   'keystone_authtoken/admin_user':        value => 'neutron';
  #   'keystone_authtoken/admin_password':    value => $neutron_user_password;
  #}

  class { '::neutron::server':

    auth_host         => $::ipaddress,
    auth_password     => $neutron_user_password,
    connection        => $sql_connection,
    sql_connection    => false,
  }

  if $neutron_core_plugin == 'neutron.plugins.ml2.plugin.Ml2Plugin' {

    neutron_config {
      'DEFAULT/service_plugins':
        value =>
join(['neutron.services.l3_router.l3_router_plugin.L3RouterPlugin',]),
    }
    ->
    class { '::neutron::plugins::ml2':
      type_drivers           => $ml2_type_drivers,
      tenant_network_types   => $ml2_tenant_network_types,
      mechanism_drivers      => $ml2_mechanism_drivers,
      flat_networks          => $ml2_flat_networks,
      network_vlan_ranges    => $ml2_network_vlan_ranges,
      tunnel_id_ranges       => $ml2_tunnel_id_ranges,
      vxlan_group            => $ml2_vxlan_group,
      vni_ranges             => $ml2_vni_ranges,
      enable_security_group => str2bool_i("$ml2_security_group"),
      firewall_driver        => $ml2_firewall_driver,
    }

    # If cisco nexus is part of ml2 mechanism drivers,
    # setup Mech Driver Cisco Neutron plugin class.
    if ('cisco_nexus' in $ml2_mechanism_drivers) {
      class { 'neutron::plugins::ml2::cisco::nexus':
        nexus_config         => $nexus_config,
      }
    }
  }

  if $neutron_core_plugin == 'neutron.plugins.cisco.network_plugin.PluginV2' {
    class { 'quickstack::neutron::plugins::cisco':
      neutron_db_password           => $neutron_db_password,
      neutron_user_password         => $neutron_user_password,
      ovs_vlan_ranges               => $ovs_vlan_ranges,
      cisco_vswitch_plugin          => $cisco_vswitch_plugin,
      nexus_config                  => $nexus_config,
      cisco_nexus_plugin            => $cisco_nexus_plugin,
      nexus_credentials             => $nexus_credentials,
      provider_vlan_auto_create     => $provider_vlan_auto_create,
      provider_vlan_auto_trunk      => $provider_vlan_auto_trunk,
      mysql_host                    => $mysql_host,
      mysql_ca                      => $mysql_ca,
      tenant_network_type           => $tenant_network_type,
    }
  }

  #class { '::neutron::plugins::ovs':
  #  sql_connection      => $sql_connection,
  #  tenant_network_type => $tenant_network_type,
  #  network_vlan_ranges => $ovs_vlan_ranges,
  #  tunnel_id_ranges    => $tunnel_id_ranges,
  #  vxlan_udp_port      => $ovs_vxlan_udp_port,
  #}

  neutron_plugin_ovs { 'AGENT/l2_population': value => "$ovs_l2_population"; }

  $local_ip = find_ip("$ovs_tunnel_network","$ovs_tunnel_iface","")

  class { '::neutron::agents::ovs':
    bridge_uplinks    => $ovs_bridge_uplinks,
    local_ip          => $local_ip,
    bridge_mappings   => $ovs_bridge_mappings,
    enable_tunneling => str2bool_i("$enable_tunneling"),
    tunnel_types      => $ovs_tunnel_types,
    vxlan_udp_port    => $ovs_vxlan_udp_port,
  }

  class { '::neutron::agents::dhcp': }

  class { '::neutron::agents::l3':
    external_network_bridge => $external_network_bridge,
  }

  class { 'neutron::agents::metadata':
    auth_password => $neutron_user_password,
    shared_secret => $neutron_metadata_proxy_secret,
    auth_url      => "http://${controller_priv_host}:35357/v2.0",
    metadata_ip   => $controller_priv_host,
  }

  if str2bool_i("$lbaas") {
    class { 'neutron::agents::lbaas': }
  }

  if str2bool_i("$fwaas") {
    class { 'neutron::services::fwaas': }
  }

  if str2bool_i("$vpnaas") {
    class { 'neutron::agents::vpnaas': }
  }

  if str2bool_i("$neutron_metering_agent") {
    class { 'neutron::agents::metering': }
  }

  class {'quickstack::neutron::firewall::gre': }

  class {'quickstack::neutron::firewall::vxlan':
    port => $ovs_vxlan_udp_port,
  }

  firewall { '001 neutron server (API)':
    proto    => 'tcp',
    dport    => ['9696'],
    action   => 'accept',
  }

}
```

6.2.2 Parameters update for Sahara, LBaas, FWaaS, and VPNaaS

```
cat /usr/share/openstack-foreman-
installer/puppet/modules/quickstack/manifests/params.pp
class quickstack::params (
# This class needs to go away.

# Logs
$admin_email                 = "admin@${::domain}",
$verbose                     = 'true',

$heat_cfn                    = 'false',
$heat_cloudwatch             = 'false',

# Passwords are currently changed to decent strings by sed
# during the setup process. This will move to the Foreman API v2
# at some point.
$admin_password              = 'CHANGEME',
$ceilometer_metering_secret  = 'CHANGEME',
$ceilometer_user_password    = 'CHANGEME',
$heat_user_password          = 'CHANGEME',
$heat_db_password            = 'CHANGEME',
$horizon_secret_key          = 'CHANGEME',
$keystone_admin_token        = 'CHANGEME',
$keystone_db_password        = 'CHANGEME',
$mysql_root_password         = 'CHANGEME',
$neutron_db_password         = 'CHANGEME',
$neutron_user_password       = 'CHANGEME',
$nova_db_password            = 'CHANGEME',
$nova_user_password          = 'CHANGEME',

# Cinder
$cinder_db_password            = 'CHANGEME',
$cinder_user_password          = 'CHANGEME',
# Cinder backend - Several backends should be able to coexist
$cinder_backend_gluster        = false,
$cinder_backend_gluster_name   = 'glusterfs_backend',
$cinder_backend_iscsi          = false,
$cinder_backend_iscsi_name     = 'iscsi_backend',
$cinder_backend_nfs            = false,
$cinder_backend_nfs_name       = 'nfs_backend',
$cinder_backend_eqlx           = false,
$cinder_backend_eqlx_name      = ['eqlx_backend'],
$cinder_multiple_backends     = false,
$cinder_backend_rbd           = false,
$cinder_backend_rbd_name      = 'rbd_backend',
# Cinder gluster
$cinder_gluster_volume        = 'cinder',
$cinder_gluster_path          = '/srv/gluster/cinder',
$cinder_gluster_peers         = [ '192.168.0.4', '192.168.0.5', '192.168.0.6' ],
$cinder_gluster_replica_count = '3',
$cinder_glusterfs_shares      = [ '192.168.0.4:/cinder -o backup-volfile-
servers=192.168.0.5' ],
# Cinder nfs
$cinder_nfs_shares            = [ '192.168.0.4:/cinder' ],
$cinder_nfs_mount_options     = '',
# Cinder Dell EqualLogic
$cinder_san_ip                = ['192.168.124.11'],
$cinder_san_login             = ['grpadmin'],
$cinder_san_password          = ['CHANGEME'],
$cinder_san_thin_provision    = [false],
$cinder_eqlx_group_name       = ['group-0'],
$cinder_eqlx_pool             = ['default'],
$cinder_eqlx_use_chap         = [false],
$cinder_eqlx_chap_login       = ['chapadmin'],
$cinder_eqlx_chap_password    = ['CHANGEME'],
#  Cinder RBD
$cinder_rbd_pool              = 'volumes',
$cinder_rbd_ceph_conf         = '/etc/ceph/ceph.conf',
$cinder_rbd_flatten_volume_from_snapshot
                              = false,
$cinder_rbd_max_clone_depth   = '5',
$cinder_rbd_user              = 'volumes',
$cinder_rbd_secret_uuid       = '',

# Glance
$glance_db_password           = 'CHANGEME',
$glance_user_password         = 'CHANGEME',
$glance_backend               = 'file',

# Glance RBD
```

```
$glance_rbd_store_user        = 'images',
$glance_rbd_store_pool        = 'images',

# Glance_Gluster
$glance_gluster_volume        = 'glance',
$glance_gluster_path          = '/srv/gluster/glance',
$glance_gluster_peers         = [ '192.168.0.4', '192.168.0.5', '192.168.0.6' ],
$glance_gluster_replica_count = '3',

# Gluster
$gluster_open_port_count      = '10',

# Networking
$neutron                       = 'false',
$controller_admin_host         = '172.16.0.1',
$controller_priv_host          = '172.16.0.1',
$controller_pub_host           = '172.16.1.1',
$nova_default_floating_pool    = 'nova',

# Nova-network specific
$fixed_network_range           = '10.0.0.0/24',
$floating_network_range        = '10.0.1.0/24',
$auto_assign_floating_ip       = 'True',

# Neutron specific
$neutron_metadata_proxy_secret = 'CHANGEME',

$mysql_host                    = '172.16.0.1',
$amqp_provider                 = 'rabbitmq',
$amqp_host                     = '172.16.0.1',
$amqp_username                 = 'openstack',
$amqp_password                 = 'CHANGEME',
$enable_ovs_agent              = 'true',
$tenant_network_type           = 'gre',
$ovs_vlan_ranges               = undef,
$ovs_bridge_mappings           = [],
$ovs_bridge_uplinks            = [],
$configure_ovswitch            = 'true',
  $enable_tunneling              = 'True',
  $ovs_vxlan_udp_port            = '4789',
  $ovs_tunnel_types              = [],

  $lbaas                  = 'false',
  $fwaas                  = 'false',
  $vpnaas                 = 'false',
  $neutron_metering_agent     = 'true',

  # neutron plugin config
  $neutron_core_plugin           = 'neutron.plugins.ml2.plugin.Ml2Plugin',
  # If using the Cisco plugin, use either OVS or n1k for virtualised l2
  $cisco_vswitch_plugin          =
'neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2',
  # If using the Cisco plugin, Nexus hardware can be used for l2
  $cisco_nexus_plugin            =
'neutron.plugins.cisco.nexus.cisco_nexus_plugin_v2.NexusPlugin',

  # If using the nexus sub plugin, specify the hardware layout by
  # using the following syntax:
  # $nexus_config = { 'SWITCH_IP' => { 'COMPUTE_NODE_NAME' : 'PORT' } },
  $nexus_config                  = undef,

  # Set the nexus login credentials by creating a list
  # of switch_ip/username/password strings as per the example below:
  $nexus_credentials             = undef,

  # provider network settings
  $provider_vlan_auto_create     = 'false',
  $provider_vlan_auto_trunk      = 'false',
  $mysql_virt_ip_nic             = '172.16.0.1',
  $mysql_virt_ip_cidr_mask       = 'MYSQL_CIDR_MASK',
  $mysql_shared_storage_device   = 'MYSQL_SHARED_STORAGE_DEVICE',
  $mysql_shared_storage_options  = '',
  # e.g. "nfs"
  $mysql_shared_storage_type     = 'MYSQL_SHARED_STORAGE_TYPE',
  $mysql_clu_member_addrs        = 'SPACE_SEPARATED_IP_ADDRS',
  $mysql_resource_group_name     = 'mysqlgroup',
  # Sahara
  $sahara               = 'false',
  $sahara_db_password          = 'CHANGEME',
  $sahara_user_password               = 'CHANGEME',
  $sahara_use_neutron          = 'true',

  # SSL
  $ssl                            = 'false',
  $freeipa                        = 'false',
  $mysql_ca                       = '/etc/ipa/ca.crt',
  $mysql_cert                     = undef,
  $mysql_key                      = undef,
  $amqp_ca                        = undef,
  $amqp_cert                      = undef,
  $amqp_key                       = undef,
  $horizon_ca                     = '/etc/ipa/ca.crt',
  $horizon_cert                   = undef,
  $horizon_key                    = undef,
  $amqp_nssdb_password            = 'CHANGEME',

  # Pacemaker
  $pacemaker_cluster_name         = 'openstack',
  $pacemaker_cluster_members      = '',
  $ha_loadbalancer_public_vip     = '172.16.1.10',
  $ha_loadbalancer_private_vip    = '172.16.2.10',
  $ha_loadbalancer_group          = 'load_balancer',
  $fencing_type                   = 'disabled',
  $fence_xvm_clu_iface            = 'eth2',
  $fence_xvm_manage_key_file      = false,
  $fence_xvm_key_file_password    = '12345678isTheSecret',
  $fence_ipmilan_address          = '10.10.10.1',
  $fence_ipmilan_username         = '',
  $fence_ipmilan_password         = '',
  $fence_ipmilan_interval         = '60s',

  # Gluster Servers
  $gluster_device1       = '/dev/vdb',
  $gluster_device2       = '/dev/vdc',
  $gluster_device3       = '/dev/vdd',
  $gluster_fqdn1         = 'gluster-server1.example.com',
  $gluster_fqdn2         = 'gluster-server2.example.com',
  $gluster_fqdn3         = 'gluster-server3.example.com',
  # One port for each brick in a volume
  $gluster_port_count    = '9',
  $gluster_replica_count = '3',
  $gluster_uuid1         = 'e27f2849-6f69-4900-b348-d7b0ae497509',
  $gluster_uuid2         = '746dc27e-b9bd-46d7-a1a6-7b8957528f4c',
  $gluster_uuid3         = '5fe22c7d-dc85-4d81-8c8b-468876852566',
  $gluster_volume1_gid   = '165',
  $gluster_volume1_name  = 'cinder',
  $gluster_volume1_path  = '/cinder',
  $gluster_volume1_uid   = '165',
  $gluster_volume2_gid   = '161',
  $gluster_volume2_name  = 'glance',
  $gluster_volume2_path  = '/glance',
  $gluster_volume2_uid   = '161',
  $gluster_volume3_gid   = '160',
  $gluster_volume3_name  = 'swift',
  $gluster_volume3_path  = '/swift',
  $gluster_volume3_uid   = '160',
) {
}
```

6.2.3 Quickstack Sahara Module

```
cat /usr/share/openstack-foreman-
installer/puppet.modules/quickstack/manifests/sahara.pp
# == Class: quickstack::sahara
#
# A class to configure Sahara




class quickstack::sahara (
  $sahara_user_password,
  $sahara_db_password,
  $sahara_use_neutron,
  $controller_admin_host,
  $controller_priv_host,
  $controller_pub_host,
  $mysql_ca,
  $mysql_host,
  $ssl,
  $verbose,
  $rpm_install                 = 'true',
  $development                 = 'false',
) {

  #if str2bool_i("$ssl")  {
  #  $sql_connection = "mysql://sahara:${sahara_db_password}@${mysql_host}/sahara?
ssl_ca=${mysql_ca}"
  #} else {
  #  $sql_connection = "mysql://sahara:${sahara_db_password}@${mysql_host}/sahara"
  #}

  class { 'sahara::db::mysql': password => $sahara_db_password, }

  class { 'sahara::keystone::auth':
    password         => $sahara_user_password,
    public_address   => $controller_pub_host,
    admin_address    => $controller_admin_host,
    internal_address => $controller_priv_host,
  }

  class { '::sahara':
    sahara_host        => $controller_priv_host,
    db_host            => $mysql_host,
    sahara_db_password => $sahara_db_password,
    keystone_auth_host => $controller_priv_host,
    keystone_password  => $sahara_user_password,
    sahara_verbose     => $verbose,
  }
  class { 'sahara::dashboard':
    sahara_host => $controller_priv_host,
    use_neutron => $sahara_use_neutron,
    rpm_install => $rpm_install,
    development => $development,
  }


}
```

6.2.4 Quickstack Controller Common Sahara Details

Add a call to Sahara:

```
vi /usr/share/openstack-foreman-
installer/puppet/modules/quickstack/manifests/controller-common.pp


...params...
  $sahara                          = $quickstack::params::sahara,
  $sahara_db_password              = $quickstack::params::sahara_db_password,
  $sahara_user_password            = $quickstack::params::sahara_user_password,
  $sahara_use_neutron              = $quickstack::params::sahara_use_neutron,


... main ...
  if (str2bool_i("$sahara")) {
    class { 'quickstack::sahara':
      sahara_user_password        => $sahara_user_password,
      sahara_db_password          => $sahara_db_password,
      sahara_use_neutron          => $sahara_use_neutron,
      controller_admin_host       => $controller_admin_host,
      controller_priv_host        => $controller_priv_host,
      controller_pub_host         => $controller_pub_host,
      mysql_ca                    => $mysql_ca,
      mysql_host                  => $mysql_host,
      ssl                         => $ssl,
      verbose                     => $verbose,
    }
    firewall { '001 sahara server (API)':
      proto    => 'tcp',
      dport    => ['8386'],
      action   => 'accept',
    }


  }
```

6.2.5 Neutron RHEL 7 libreswan update

I modified the /usr/share/openstack-puppet-modules/neutron/manifests/params.pp to include libreswan for RHEL 7 rather than openswan. The code snippet for this is as follows:

```
    if $::operatingsystemrelease =~ /^7.*/ {
      $openswan_package = 'libreswan'
    } else {
      $openswan_package = 'openswan'
    }
```

6.2.6 Sahara RPM install update

I updated the /usr/share/openstack-puppet-modules/sahara/manifests/params.pp to force an RPM install:

```
   rpm_install = true
```

In addition to this I modified the sahara/manifests/db/mysql.pp to comment out the mysql work that is done elsewhere:

```
#class mysql::bindings::python {
#   include mysql::params
#   package { 'python-mysqldb':
#     ensure   => $mysql::params::python_package_ensure,
#     name     => $mysql::params::python_package_name,
#     provider => $mysql::params::python_package_provider,
#   }
#}
    require  => Class['mysql::config'],
```

And sahara/manifests/dashboard.pp:

```
  #if !defined(Package['python-pip']) {
```

```
#  package { 'python-pip': ensure => latest, }
#}

if $sahara::params::development {
  info('Installing the developement version of sahara dashboard')

  if !defined(Package['python-pip']) {
    package { 'python-pip': ensure => latest, }
  }
```

And sahara/manifests/install.pp

```
if $sahara::params::development {
  info("Installing and using the sahara development version. URL:
    ${sahara::params::development_build_url}")

# this is here until this fix is released
# https://bugs.launchpad.net/ubuntu/+source/python-pbr/+bug/1245676

  if !defined(Package['git']) {
    package { 'git': ensure => latest, }
  }

  if !defined(Package['python-pip']) {
    package { 'python-pip':
      ensure  => latest,
      require => Package['git']
    }
  }

  if $::osfamily == 'Debian' {
    if !defined(Package['python-dev']) {
      package { 'python-dev':
        ensure  => latest,
        require => Package['python-pip']
      }
    }
  } elsif $::osfamily == 'Redhat' {
    if !defined(Package['python-devel']) {
      package { 'python-devel':
        ensure  => latest,
        require => Package['python-pip']
      }
    }
    if !defined(Package['python-jinja2']) {
      package { 'python-jinja2':
        ensure  => latest,
        require => Package['python-pip']
      }
    }
  }
  package { 'sahara':
    ensure   => installed,
    provider => pip,
    source   => $sahara::params::development_build_url,
    require  => Package['python-pip'],
  }
```