# Reference Architectures 2017

# Deploying Red Hat OpenStack Platform 10 on HPE ProLiant DL Servers

Ken Bell

# Reference Architectures 2017 Deploying Red Hat OpenStack Platform 10 on HPE ProLiant DL Servers

Ken Bell
refarch-feedback@redhat.com

## Legal Notice



## Abstract

The purpose of this document is to provide guidelines and considerations for installing and configuring Red Hat Openstack Platform 10 on HPE DL ProLiant Servers.

# Table of Contents

# COMMENTS AND FEEDBACK

In the spirit of open source, we invite anyone to provide feedback and comments on any reference architecture. 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.

# CHAPTER 1. EXECUTIVE SUMMARY

Many organizations are implementing private clouds to streamline operations by providing end users with self-service portals where they can provision and maintain virtual machines and application services. OpenStack® is rapidly becoming the standard for private cloud deployments; providing support for heterogeneous software, hardware, and operating systems. OpenStack can provide organizations with a platform that can deliver Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) offerings to end user and developer communities. This reference architecture will provide an example of an entry level Red Hat OpenStack Platform 10 deployment on industry leading Hewlett Packard Enterprise (HPE) ProLiant DL servers. This reference architecture can be used as an installation and deployment example for organizations deploying their first Red Hat OpenStack Platform 10 based private cloud.

**Intended Audience**

This document is intended for systems engineers, systems administrators, architects, and installers responsible for installing and maintaining OpenStack private cloud environments. The reader of this document should be familiar with Red Hat OpenStack Platform, Red Hat Enterprise Linux®, HPE ProLiant servers, and networking equipment.

# CHAPTER 2. SOFTWARE OVERVIEW

### Red Hat OpenStack Platform 10

Red Hat OpenStack Platform 10 is based on the Newton OpenStack release. The Newton OpenStack release is the 14th OpenStack release. This release improves reliability and provides new features and functionality.

### New Features in Red Hat OpenStack Platform 10

There are many new features available in the Red Hat OpenStack Platform 10. Just a few of these features are listed below:

- Monitoring – A Sensu client is installed on each of the nodes to allow for monitoring of the individual OpenStack nodes and services.
- Graphical Installer UI – The deployment of the Red Hat OpenStack Platform overcloud can now be performed using a Graphical User Interface (GUI).
- Composable Roles – OpenStack services can now be combined or segregated to create customized roles for OpenStack nodes.
- Director - Support as a virtual machine, with this release of Red Hat OpenStack Platform 10 the Red Hat OpenStack Platform 10 director is now supported to be deployed on a virtual machine.

### Red Hat Ceph Storage

OpenStack private clouds leverage both block and object storage. Ceph is the preferred storage platform to provide these services.

Red Hat OpenStack Platform 10 also includes new Tech Preview features that are available for testing.

For a full list of new features and features that are still in Technology Preview, please refer to the Red Hat OpenStack Platform 10 release notes.

### HPE ProLiant Service Pack

HPE ProLiant server firmware and software updates are maintained in the HPE ProLiant Service Pack. The servers were updated with the latest HPE ProLiant Service Pack, SPP 2016.10.0 Table 1, below shows the firmware versions installed on the ProLiant DL servers used in this reference architecture.

| Firmware | Version |
|---|---|
| iLO | 2.50 Sep 23 2016 |
| System ROM | P89 v2.30 (09/13/2016) |
| Redundant System ROM | P89 v2.00 (12/27/2015) |

| Firmware | Version |
|---|---|
| HP Ethernet 1Gb 4-port 331i Adapter - NIC | 17.4.41 |
| HP Ethernet 10Gb 2-port 560FLR-SFP+ Adapter | 1.1200.0 |
| HPE Ethernet 10Gb 2-port 562SFP+ Adapter | 1.1200.0 |
| HPE Smart Storage Battery 1 Firmware | 1.1 |
| Intelligent Platform Abstraction Data | 23.05 |
| Intelligent Provisioning | N/A |
| Power Management Controller Firmware | 1.0.9 |
| Power Management Controller FW Bootloader | 1.0 |
| SAS Programmable Logic Device | Version 0x02 |
| Server Platform Services (SPS) Firmware | 3.1.3.21.0 |
| Smart Array P440ar Controller | 4.52 |
| System Programmable Logic Device | Version 0x34 |

*Table 1: Firmware*

ProLiant Service Pack can be found at:
http://h17007.www1.hpe.com/us/en/enterprise/servers/products/service_pack/spp/index.aspx

# CHAPTER 3. HARDWARE

The server hardware used in this reference architecture are Hewlett Packard Enterprise (HPE) rack mounted servers, specifically the HPE ProLiant DL360 Gen9 and HPE ProLiant DL380 Gen9 models. The network switches used in this reference architecture are HPE FlexFabric switches, the HPE FF 5930 and HPE FF 5700. The following section provides an overview of the technical specifications for the servers and network equipment used in this reference architecture.

The servers and networking components are shown below in Figure 1. There are two top of rack switches, an HPE FF 5700 and an HPE FF 5930, the HPE FF 5700 is used for management (iLO) and provisioning, and the HPE FF 5930 is used for OpenStack networking including; internal API, external, storage, storage management, and tenant communications. There are eleven rack mounted servers shown in Figure 1; one Red Hat OpenStack Platform director, three Red Hat OpenStack Platform controllers, four Red Hat OpenStack Platform compute nodes, and three Red Hat Ceph Storage nodes.

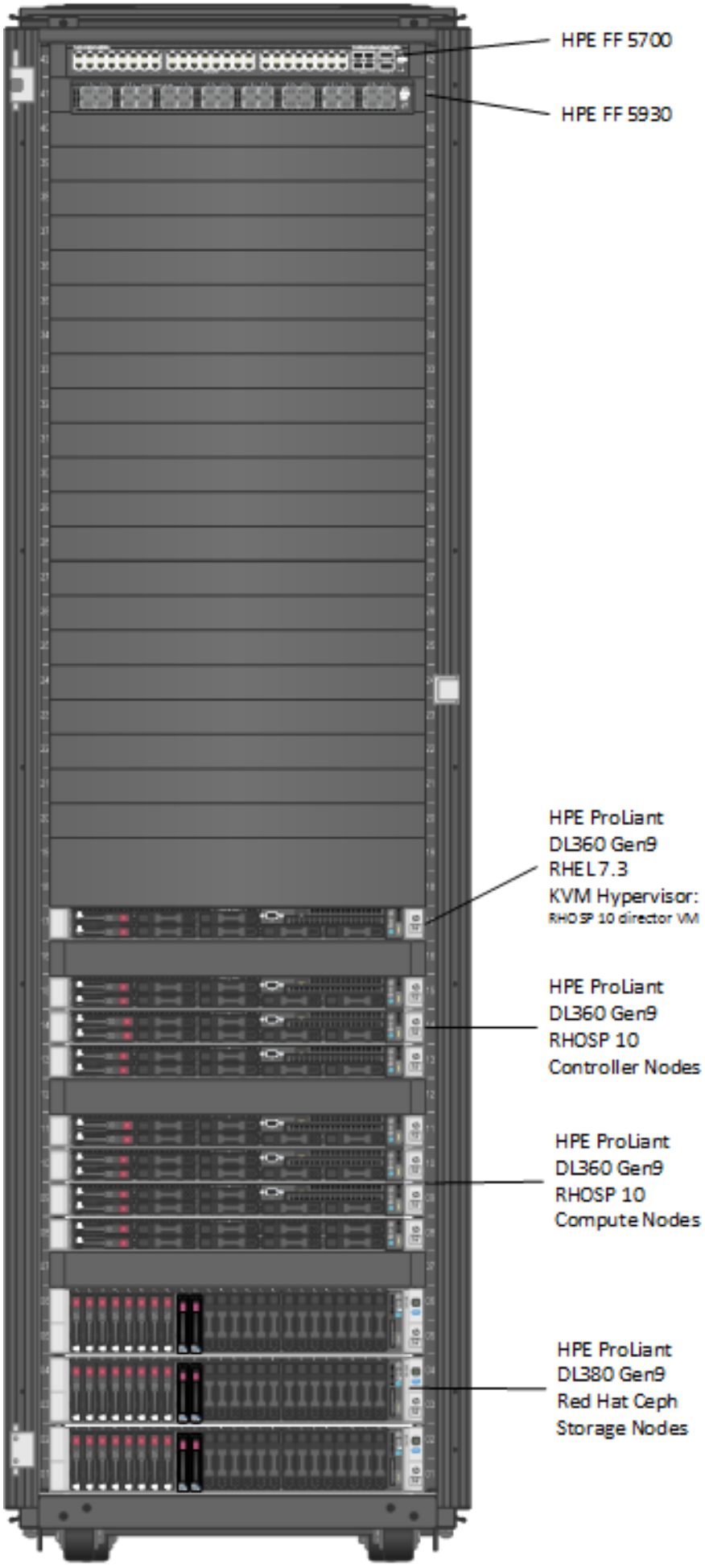



*Figure 1: HPE ProLiant DL Rack*

## 3.1. HPE PROLIANT SERVERS

### 3.1.1. HPE ProLiant DL380 Gen9 Server

The HPE ProLiant DL380 Gen9 is an industry leader in rack mount servers. This reference architecture will leverage the HPE DL380 Gen9 server for a three node Red Hat Ceph Storage cluster to provide highly available block and object storage for the Red Hat OpenStack Platform 10 environment.

**Specifications:**

- Processor family Intel Xeon E5-2600 v3 product family; Intel Xeon E5-2600 v4 product family
- Processor core available 22, 20, 18, 16, 14, 12, 10, 8, 6, or 4
- Maximum memory, 3.0TB; With 128-GB DDR4
- Storage controller (1) Dynamic Smart Array B140i and/or; (1,) Smart Array P440ar; (1) Smart Array P840; Depending on model
- Form factor (fully configured) 2U

Refer to the following link for complete HPE ProLiant DL380 Gen9 server specifications: https://www.hpe.com/us/en/product-catalog/servers/proliant-servers/pip.specifications.hpe-proliant-dl380-gen9-server.7271241.html

**Red Hat Ceph Storage Nodes HPE ProLiant DL380 Gen9 Configuration**

The Red Hat Ceph Storage nodes are comprised of three HPE ProLiant DL380 Gen9 servers with the following configuration;

- CPU – 1 x Intel Xeon E5-2643v3 (3.4GHz/6-core/20MB/135W)
- Memory – 64GB Single Rank x4 DDR4-2133
- Storage - 12 HP 1.2TB 6G SAS 10K rpm SFF Drives and 2 HP 400GB 12G SAS Write Intensive SFF Solid State Drives
- Network - HP Ethernet 10Gb 2-port 560FLR-SFP Adapter and ALOM 2 port 10Gb NIC

### 3.1.2. HPE ProLiant DL360 Gen9 Server

The HPE ProLiant DL360 Gen9 is a 1U rack mount server that will host the Red Hat OpenStack Platform controller, compute, and director nodes. In this reference architecture the Red Hat OpenStack Platform 10 director will be installed as virtual machine running on a Red Hat Enterprise Linux 7.3 KVM host. The Red Hat OpenStack Platform director could also be installed as a dedicated baremetal host.

**Specifications:**

- Processor family Intel Xeon E5-2600 v3 product family; Intel Xeon E5-2600 v4 product family
- Processor core available 22, 20, 18, 16, 14, 12, 10, 8, 6, or 4
- Maximum memory 1.5TB
- Storage controller (1) Dynamic Smart Array B140i or; (1) H240ar Host Bus Adapter or; (1) Smart Array P440ar; Depending on model

- Form factor (fully configured) 1U

Refer to the following link for complete HPE ProLiant DL360 Gen9 server specifications: https://www.hpe.com/us/en/product-catalog/servers/proliant-servers/pip.hpe-proliant-dl360-gen9-server.7252836.html

**KVM Hypervisor ProLiant DL360 Gen9 Configuration**

- Server Platform – HPE ProLiant DL360 Gen9
- CPU – 2 x Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W)
- Memory – 64GB Dual Rank x4 DDR4-2133
- Storage – 2 x HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) Drives
- Network - HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter and ALOM 2 port 10Gb NIC

**Control Plane HPE ProLiant DL360 Gen9 Configuration**

The control plane is deployed on three physical HPE ProLiant DL360 Gen9 servers with the following configuration:

- CPU - 2 x Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W)
- Memory - 64GB Dual Rank x4 DDR4-2133
- Storage - 2 x HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) Drives
- Network - HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter and ALOM 2 port 10Gb NIC

**Compute Plane HPE ProLiant DL360 Gen9 Configuration**

The compute plane is comprised of four HPE ProLiant DL360 Gen9 servers.

- CPU - 2 x Intel Xeon E5-2690v3 (2.6GHz/12-core/30MB/135W) Processor
- Memory - 256GB Dual Rank x4 DDR4-2133
- Storage - 2 x HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) Drives
- Network - HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter and ALOM 2 port 10Gb NIC

## 3.2. HPE NETWORK

### 3.2.1. HPE FlexFabric 5930 32QSFP+ Switch

This solution uses the HPE FlexFabric 5930 switch for Red Hat OpenStack Platform External, Storage, Storage Mangement, Tenant, and Internal API network communications. The HPE specification overview for this switch is described below:
"HPE FlexFabric 5930 Series switches are high density, ultra low latency 10 Gigabit Ethernet (GbE) and 40 GbE top of rack (ToR) datacenter switches. This 1U high model has 32x QSFP+ 40GbE ports, 2x power supply slots, and 2x fan tray slots. Power supplies and fan trays must be ordered separately. The FlexFabric 5930 Switch Series is ideally suited for deployment at the aggregation or server access layer of large enterprise data centers, or at the core layer of medium sized enterprises. These are optimized to meet the increasing requirements for higher performance server connectivity, convergence of Ethernet and storage traffic, the capability to handle virtual environments, and ultra low latency. The FlexFabric product line offers modular core, aggregation, top of rack switching for datacenters."

This description and the full specifications for the HPE FF 5930 can be found at the following link: https://www.hpe.com/us/en/product-catalog/networking/networking-switches/pip.overview.networking-switches.7526493.html

### 3.2.2. HPE FlexFabric 5700 48G 4XG 2QSFP+ Switch

The iLO management and Red Hat OpenStack Platform provisioning networks use the HPE FlexFabric 5700 switch. The HPE specifications overview for the HPE FF 5700 is switch described below:
"HPE FlexFabric 5700 Series switches are cost effective, high density, ultra low latency, top of rack (ToR) data center switches. This model comes with 48x 10/100/1000 ports, 4x fixed 1000 / 10000 SFP+ ports, and 2x QSFP+ for 40 GbE connections. They are ideally suited for deployment at the server access layer of large enterprise data centers. The FlexFabric product line offers modular core, aggregation, top of rack switching for data centers."

This description and the full specifications for the HPE FF 5930 can be found at the following link: https://www.hpe.com/us/en/product-catalog/networking/networking-switches/pip.overview.networking-switches.6638103.html

# CHAPTER 4. SOLUTION DESIGN

This solution is comprised of eleven HPE ProLiant DL servers; eight HPE ProLiant DL360 Gen9 and three HPE ProLiant DL380 Gen9 servers. The servers are configured to create a highly available platform for a Red Hat OpenStack Platform 10 deployment. Each server is configured with redundant hard drives (operating system only), redundant fans, and redundant power supplies. The OpenStack Internal API, Storage, Storage Management, External, and Tenant networks are configured to run on bonded ten Gigabit interfaces that are split across two network interface cards. Hardware based RAID1 provides increased availability for the Red Hat OpenStack Platform 10 and Red Hat Ceph Storage server nodes operating system disks. The three node Red Hat OpenStack Platform 10 controller deployment is configured for high availability clustering using Pacemaker and HA Proxy. The three node Red Hat Ceph Storage cluster provides a high availability storage platform for OpenStack block and object storage services. Red Hat OpenStack Platform 10 services are monitored using a preinstalled Sensu client which can be configured to automatically connect to an existing Sensu monitoring host during deployment.

## 4.1. RED HAT OPENSTACK PLATFORM 10 DIRECTOR

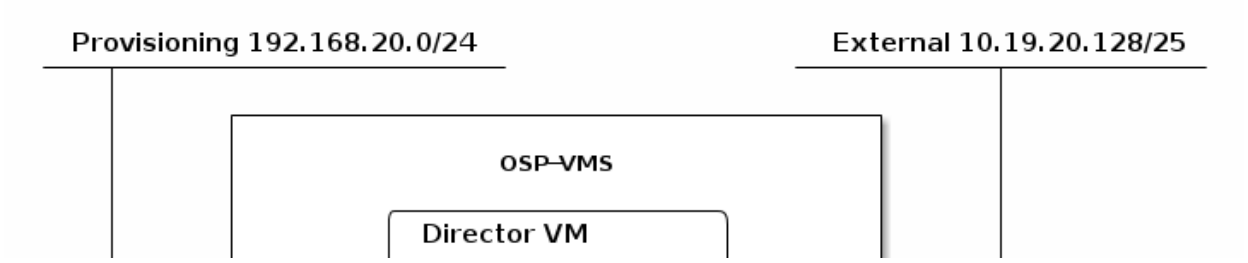
In this reference architecture the Red Hat OpenStack Platform 10 director is installed as a virtual machine on one of the HPE DL360 Gen9 servers. The host is running Red Hat Enterprise Linux 7.3 with KVM. The Red Hat OpenStack Platform 10 director virtual machine is running Red Hat Enterprise Linux 7.3. By running the Red Hat OpenStack Platform 10 director as a virtual machine, the physical server can be used to support additional services and virtual machines. Additionally, running the Red Hat OpenStack Platform director on a virtualization platform provides the ability to snapshot the virtual machine at various stages of the installation. Virtual machine snaphots can also be useful if it is necessary to revert the system to a previously known good state or configuration. In this example there are three additional Red Hat Enterprise Linux 7.3 based virtual machines; a logging vm, and two Sensu monitoring virtual machines.

## 4.2. NETWORK CONFIGURATION

The network infrastructure is comprised of HPE FF 5700 and HPE FF 5930 network switches. The HPE FF 5700 is used for the management (iLO) and provisioning networks. The HPE FF 5930 is used for the following Red Hat OpenStack Platform 10 networks:

- External
- Internal API
- Tenant
- Storage Management
- Storage network

An overview of the network connections to each of the individual servers is shown in Figure 2 below.



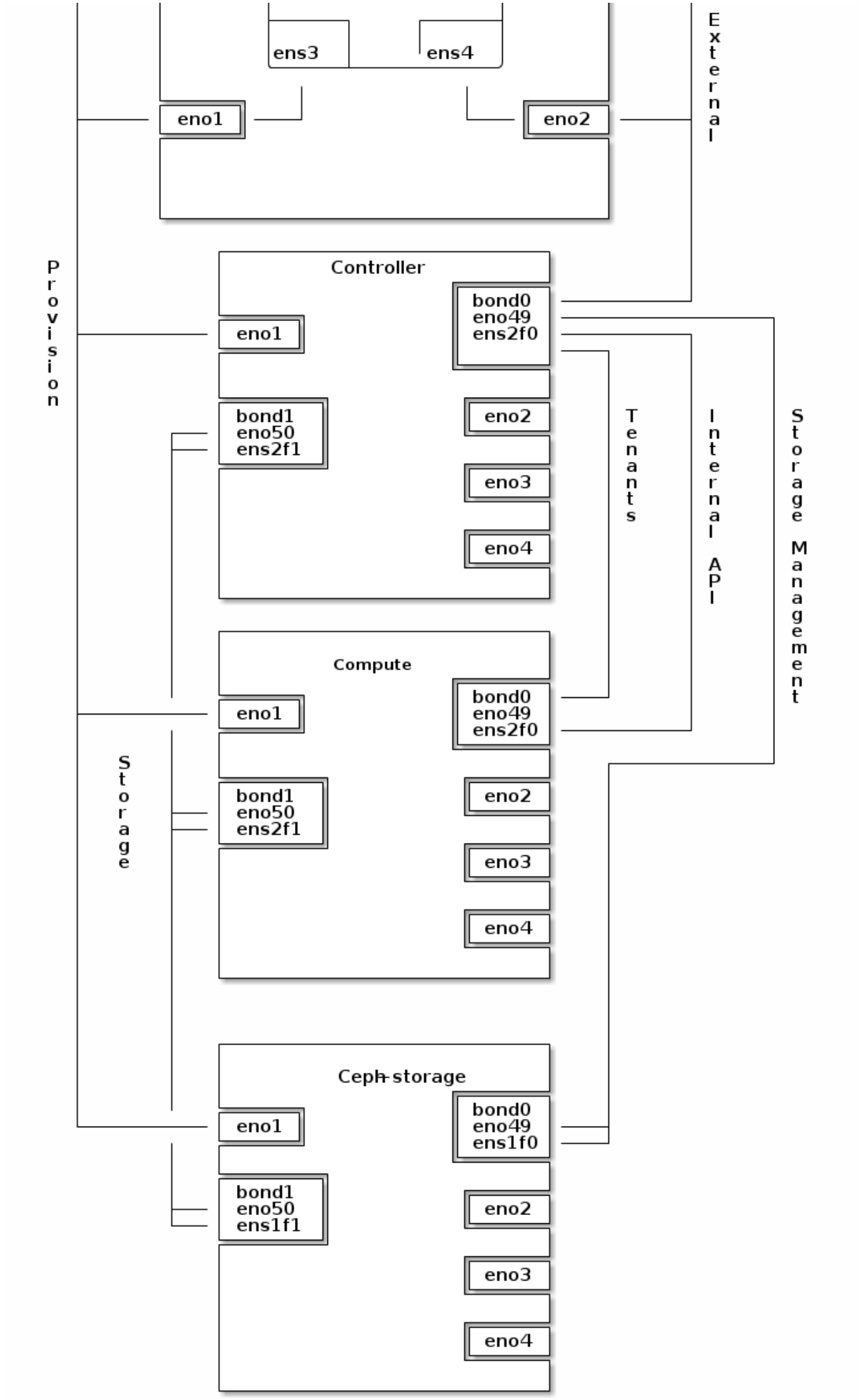

ens3
ens4
eno1
eno2
External
Provision
Controller
bond0
eno49
ens2f0
eno1
bond1
eno50
ens2f1
eno2
eno3
eno4
Tenants
Internal API
Storage Management
Compute
bond0
eno49
ens2f0
eno1
Storage
bond1
eno50
ens2f1
eno2
eno3
eno4
Ceph-storage
bond0
eno49
ens1f0
eno1
bond1
eno50
ens1f1
eno2
eno3
eno4

*Figure 2: Red Hat OpenStack Platform 10 Overview*

### 4.2.1. VLAN Configuration

A total of 7 VLANs have been configured for this solution. The VLANs include: 1(default), 104, 3040-3044.

Table 2, below lists each VLAN including the correspondingRed Hat OpenStack Platform Network, name, VLAN ID, and IP subnet.

| Red Hat OpenStack Platform Networks | VLAN Name | VLAN ID | Subnet / CIDR |
|---|---|---|---|
| External | hpe-ext | 104 | 10.19.20.128/25 |
| Provision | hpe-prov | 3040 | 192.168.20.0/24 |
| Internal API | hpe-api | 3041 | 172.16.10.0/24 |
| Tenant | hpe-ten | 3044 | 172.16.160.0/24 |
| Storage | hpe-stor | 3042 | 172.16.6.10/24 |
| Storage Mgmt | hpe-stormgmt | 3043 | 172.16.16.0/24 |

*Table 2: Red Hat OpenStack Platform Network and VLAN Assignments*

### 4.2.2. HPE FF 5930 Link Aggregation and Bonding

The HPE FF 5930 is configured with 32 QSFP 40 Gigabit ports. Each port will be split into four 10 Gigabit network ports and connected to 10 Gigabit network interfaces on the controller, compute, and ceph storage nodes. Each Red Hat OpenStack Platform 10 compute, controller, and Red Hat Ceph Storage node is equipped with two dual port 10 Gigabit Network Interface Cards. The four 10 Gigabit interfaces on each node will form two bonds on each server, one for storage containing the Storage network, and the other for OpenStack cloud communications including the Internal API, External, Storage Management, and Tenant networks. These bonded interfaces will be connected to Link Aggregation groups on the HPE FF 5930 switch which have the corresponding VLANs assigned. In this reference architecture there is only one HPE FF 5930, for additional redundancy a second HPE FF 5930 could be added to the configuration and configured in an Intelligent Resilient Framework (IRF). The bridge aggregations and bonded interfaces could then be distributed across the two physical switches.

Storage Network

- Storage

Cloud Networks

- External
- Tenant
- Internal API
- Storage Management

### 4.2.3. Split the 40 Gigabit QSFP ports into 10 Gigabit ports

As mentioned previously, the HPE FF 5930 used in this reference architecture is configured with 32 Forty Gigabit QSFP ports. Each 40 Gigabit port will be split into four 10 Gigabit ports. The first four 40 Gigabit ports (FortyGigE 1/0/1 - FortyGigE 1/0/4) cannot be split. The first port that can be split is port 5 (FortyGigE 1/0/5). Once the interface has been split, there will be four 10 Gigabit ports labeled:

- XGE1/0/5:1 UP 10G(a) F(a) T 1
- XGE1/0/5:2 UP 10G(a) F(a) T 1
- XGE1/0/5:3 UP 10G(a) F(a) T 1
- XGE1/0/5:4 UP 10G(a) F(a) T 1

The following commands are used to split the FortyGigE interface:

```
[5930-01] interface FortyGigE 1/0/5
[5930-01-FortyGigE1/0/5] using tengige
```

Repeat for interfaces FortyGigE 1/0/6 – FortyGigE 1/0/15

This will create the Ten Gigabit interfaces shown below in Table 3:

| 10Gb Interface 1 | 10Gb Interface 2 | 10Gb Interface 3 | 10Gb Interface 4 |
|---|---|---|---|
| XGE1/0/5:1 | XGE1/0/5:2 | XGE1/0/5:3 | XGE1/0/5:4 |
| XGE1/0/6:1 | XGE1/0/6:2 | XGE1/0/6:3 | XGE1/0/6:4 |
| XGE1/0/7:1 | XGE1/0/7:2 | XGE1/0/7:3 | XGE1/0/7:4 |
| XGE1/0/8:1 | XGE1/0/8:2 | XGE1/0/8:3 | XGE1/0/8:4 |

| 10Gb Interface 1 | 10Gb Interface 2 | 10Gb Interface 3 | 10Gb Interface 4 |
|---|---|---|---|
| XGE1/0/9:1 | XGE1/0/9:2 | XGE1/0/9:3 | XGE1/0/9:4 |
| XGE1/0/10:1 | XGE1/0/10:2 | XGE1/0/10:3 | XGE1/0/10:4 |
| XGE1/0/11:1 | XGE1/0/11:2 | XGE1/0/11:3 | XGE1/0/11:4 |
| XGE1/0/12:1 | XGE1/0/12:2 | XGE1/0/12:3 | XGE1/0/12:4 |
| XGE1/0/13:1 | XGE1/0/13:2 | XGE1/0/13:3 | XGE1/0/13:4 |
| XGE1/0/14:1 | XGE1/0/14:2 | XGE1/0/14:3 | XGE1/0/14:4 |
| XGE1/0/15:1 | XGE1/0/15:2 | XGE1/0/15:3 | XGE1/0/15:4 |

*Table 3: 10Gb Interfaces*

### 4.2.4. Create the Link Aggregation Groups, Assigning Interfaces and VLANS

Link Aggregation interfaces are configured on the HPE FF 5930 switch and the corresponding VLAN have been assigned to these interfaces.

Cloud Network example:

```
[5930-01]interface Bridge-Aggregation11
 port link-type trunk
 port trunk permit vlan 1 104 3041 3043 to 3044
 link-aggregation mode dynamic
 lacp edge-port
[5930-01]interface Ten-GigabitEthernet1/0/5:1
 port link-mode bridge
 port link-type trunk
 port trunk permit vlan 1 104 3041 3043 to 3044
 port link-aggregation group 11
[5930-01]interface Ten-GigabitEthernet1/0/5:3
 port link-mode bridge
 port link-type trunk
 port trunk permit vlan 1 104 3041 3043 to 3044
 port link-aggregation group 11
```

Storage Network example:

```
[5930-01]interface Bridge-Aggregation12
 port link-type trunk
 port trunk permit vlan 1 3042
 link-aggregation mode dynamic
 lacp edge-port
[5930-01]interface Ten-GigabitEthernet1/0/5:2
 port link-mode bridge
 port link-type trunk
 port trunk permit vlan 1 3042
 port link-aggregation group 12
[5930-01]interface Ten-GigabitEthernet1/0/5:4
 port link-mode bridge
 port link-type trunk
 port trunk permit vlan 1 3042
 port link-aggregation group 12
```

The tables below, table 4 and table 5, show the full list of Bridge Aggregations, Interfaces, and VLANs defined on the HPE FF 5930 for the Red Hat OpenStack Platform 10 Cloud and Storage networks.

| Cloud Network Aggregations | Interface 1 | Interface 2 | VLANS |
|---|---|---|---|
| Bridge-Aggregation11 | Ten-GigabitEthernet1/0/5:1 | Ten-GigabitEthernet1/0/5:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation21 | Ten-GigabitEthernet1/0/6:1 | Ten-GigabitEthernet1/0/6:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation31 | Ten-GigabitEthernet1/0/7:1 | Ten-GigabitEthernet1/0/7:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation41 | Ten-GigabitEthernet1/0/8:1 | Ten-GigabitEthernet1/0/8:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation51 | Ten-GigabitEthernet1/0/9:1 | Ten-GigabitEthernet1/0/9:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation61 | Ten-GigabitEthernet1/0/10:1 | Ten-GigabitEthernet1/0/10:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation71 | Ten-GigabitEthernet1/0/11:1 | Ten-GigabitEthernet1/0/11:3 | 1,104,3041,3043-3044 |

| Cloud Network Aggregations | Interface 1 | Interface 2 | VLANS |
|---|---|---|---|
| Bridge-Aggregation81 | Ten-GigabitEthernet1/0/12:1 | Ten-GigabitEthernet1/0/12:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation91 | Ten-GigabitEthernet1/0/13:1 | Ten-GigabitEthernet1/0/13:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation101 | Ten-GigabitEthernet1/0/14:1 | Ten-GigabitEthernet1/0/14:3 | 1,104,3041,3043-3044 |
| Bridge-Aggregation111 | Ten-GigabitEthernet1/0/15:1 | Ten-GigabitEthernet1/0/15:3 | 1,104,3041,3043-3044 |

*Table 4: Cloud Network Bridge Aggregations*

The HPE FF 5930 Bridge Aggregations, Interfaces, and corresponding VLAN assignments are shown below in Table 5.

| Storage Trunk Aggregations | Interface 1 | Interface 2 | VLANS |
|---|---|---|---|
| Bridge-Aggregation12 | Ten-GigabitEthernet1/0/5:2 | Ten-GigabitEthernet1/0/5:4 | 1,3042 |
| Bridge-Aggregation22 | Ten-GigabitEthernet1/0/6:2 | Ten-GigabitEthernet1/0/6:4 | 1,3042 |
| Bridge-Aggregation32 | Ten-GigabitEthernet1/0/7:2 | Ten-GigabitEthernet1/0/7:4 | 1,3042 |
| Bridge-Aggregation42 | Ten-GigabitEthernet1/0/8:2 | Ten-GigabitEthernet1/0/8:4 | 1,3042 |
| Bridge-Aggregation52 | Ten-GigabitEthernet1/0/9:2 | Ten-GigabitEthernet1/0/9:4 | 1,3042 |

| Storage Trunk Aggregations | Interface 1 | Interface 2 | VLANS |
|---|---|---|---|
| Bridge-Aggregation62 | Ten-GigabitEthernet1/0/10:2 | Ten-GigabitEthernet1/0/10:4 | 1,3042 |
| Bridge-Aggregation72 | Ten-GigabitEthernet1/0/11:2 | Ten-GigabitEthernet1/0/11:4 | 1,3042 |
| Bridge-Aggregation82 | Ten-GigabitEthernet1/0/12:2 | Ten-GigabitEthernet1/0/12:4 | 1,3042 |
| Bridge-Aggregation92 | Ten-GigabitEthernet1/0/13:2 | Ten-GigabitEthernet1/0/13:4 | 1,3042 |
| Bridge-Aggregation102 | Ten-GigabitEthernet1/0/14:2 | Ten-GigabitEthernet1/0/14:4 | 1,3042 |
| Bridge-Aggregation112 | Ten-GigabitEthernet1/0/15:2 | Ten-GigabitEthernet1/0/15:4 | 1,3042 |

*Table 5: Storage Network Bridge Aggregations*

## 4.3. CONFIGURE THE RED HAT CEPH STORAGE NODES

The Red Hat Ceph Storage cluster in this solution is comprised of three HPE ProLiant DL380 Gen9 servers. A three node cluster is the absolute minimum replicated cluster size. A cluster size of five nodes for replication and seven nodes for erasure coding would provide better availability and increased performance.

**Disk Configuration**

The HPE ProLiant DL380 Gen9 ceph-storage nodes are configured with twelve 1.2TB SAS hard drives and two 400GB SAS SSD drives. Two of the 1.2TB drive are configured as a RAID 1 array for the operating system, the remaining ten 1.2TB SAS drives are configured into ten individual RAID 0 logical drives and to be used for OSD. The two SSD drives are configured into individual RAID 0 logical drives to be used for journal files. Create the arrays and logical drives using the HP Smart Storage Administrator. Access the HPE System Utilities by pressing **F9** during the boot process and select System Configuration, select the Smart Array P440ar controller and select Exit and launch the HP Smart Storage Administrator (HPSSA). When creating the logical drive for the operating system, take note of the Drive Unique ID located in the logical drive properties. The Drive Unique ID value will be used to identify the boot disk by cross checking the Drive Unique ID value with the serial number in the Swift data generated by the ironic introspection process.

# CHAPTER 5. DEPLOYING THE UNDERCLOUD AND OVERCLOUD

This section provides the installation and configuration details for installing both the Red Hat OpenStack Platform 10 undercloud and overcloud. Specifically, the installation and configuration details are provided for installing the Red Hat OpenStack Platform 10 director, customizing the environment configuration files (heat templates) for the overcloud deployment, and deploying the overcloud. Customization of the Red Hat OpenStack Platform 10 heat templates is performed to provide environment specific details for the overcloud deployment. Additionally, configuration details for the new monitoring and logging environments are provided in this section.

## 5.1. INSTALLATION AND CONFIGURATION OF THE RED HAT OPENSTACK PLATFORM 10 DIRECTOR

Red Hat OpenStack Platform 10 director can be installed on a bare metal machine or a virtual machine. The system resource requirements are similar for a virtual machine or a physical bare metal server. The following are the minimum system configuration requirements.

- 8 CPU cores
- 16GB RAM
- 40GB Disk Storage
- Two Network Interfaces with a minimum of 1GB, 10GB is recommended for the Provisioning Network

**Create the Red Hat OpenStack Platfom 10 director Virtual Machine**

In this reference architecture Red Hat OpenStack Platform 10 director is installed on a virtual machine. The virtual machine is hosted on a HPE ProLiant DL360 Gen9 server running Red Hat Enterprise Linux 7.3 with KVM enabled. Create a virtual machine with the following requirements.

- CPU – 8 virtual CPU cores
- Memory – 16,384 MiB Ram
- Storage – 100 GiB virtual disk
- Two Network adapters – e1000
- Software Selection – Infrastructure Server

Connect one network adapter to the Provisioning Network and the other to the External Network as shown in Figure 2.

### 5.1.1. Install and Configure the Operating System

Install Red Hat Enterprise Linux 7.3 on the newly created virtual machine that will be used for Red Hat OpenStack Platform 10 director. The following section will provide a high level overview of the Red Hat OpenStack Platform 10 Undercloud installation and configuration. This overview is specific to the reference architecture described in this document. Please refer to the Red Hat OpenStack Platform 10 installation guide for complete installation documentation.

**Create a stack user and assign a password**

```
# useradd stack
# passwd stack (specify password)
```

**Add user stack to sudoer file**

```
# echo "stack ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/stack
# chmod 0440 /etc/sudoers.d/stack
```

**Create template and image directories**

```
# su - stack
$ mkdir images
$ mkdir templates
```

**Configure hostname**

```
$ sudo hostnamectl set-hostname <RHOSP director FQDN>
director.example.com
$ sudo hostnamectl set-hostname --transient <RHOSP director FQDN>
director.example.com
```

Add the Red Hat OpenStack Platform director FQDN to /etc/hosts

**Register and attach repositories**

```
$ Register and attach repositories
$ sudo subscription-manager register
$ sudo subscription-manager list --available --all
$ sudo subscription-manager attach --pool=<pool_id>
$ sudo subscription-manager repos --disable=*
$ sudo subscription-manager repos --enable=rhel-7-server-rpms --
enable=rhel-7-server-extras-rpms --enable=rhel-7-server-rh-common-rpms --
enable=rhel-ha-for-rhel-7-server-rpms --enable=rhel-7-server-openstack-10-
rpms
$ sudo yum update -y
$ sudo reboot
```

**Install the tripleoclient**

```
$ sudo yum install -y python-tripleoclient
```

**Copy and edit the undercloud.conf file**

Installing the python-tripleoclient provides a sample *undercloud.conf.sample* file in */usr/share/instack-undercloud/*. Copy this file to */home/stack/* and modify for the Red Hat OpenStack Platform undercloud deployment. Below is an example of the modifications made to the *undercloud.conf* file for the Red Hat OpenStack Platform deployment documented in this reference architecture.

```
undercloud_hostname = director.hpecloud.lab.eng.bos.redhat.com
local_ip = 192.168.20.20/24
network_gateway = 192.168.20.20
undercloud_public_vip = 192.168.20.2
undercloud_admin_vip = 192.168.20.3
undercloud_service_certificate = /etc/pki/instack-certs/undercloud.pem
local_interface = ens4
```

```
network_cidr = 192.168.20.0/24
masquerade_network = 192.168.20.0/24
dhcp_start = 192.168.20.50
dhcp_end = 192.168.20.99
inspection_interface = br-ctlplane
inspection_iprange = 192.168.20.100,192.168.20.120
generate_service_certificate = true
certificate_generation_ca = local
undercloud_debug = false
```

### 5.1.2. Install the Undercloud

Once the tripleoclient has been installed and the *undercloud.conf* file has been modified the undercloud can be installed. Execute the following command to install the undercloud:

```
$ openstack undercloud install
```

When the installation script completes source the stackrc file and verify the undercloud is operational

```
$ openstack service list
```

### 5.1.3. Configure the Undercloud

**Install, copy, and extract the image files**

```
$ sudo yum install rhosp-director-images rhosp-director-images-ipa
```

```
$ for i in /usr/share/rhosp-director-images/overcloud-full-latest-
10.0.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-
10.0.tar; do tar -xvf $i; done
```

The following image files will extracted to */home/stack/images/*

- overcloud-full.qcow2
- overcloud-full.initrd
- overcloud-full.vmlinuz
- ironic-python-agent.initramfs
- ironic-python-agent.kernel

**Modify image files (optional)**

The overcloud image can be modified using virt-customize to add a specific password for the root user. This is an optional step, but can be useful when troubleshooting the overcloud deployment.

```
$ sudo systemctl start libvirtd
$ virt-customize -a overcloud-full.qcow2 --root-password password:redhat
--run-command 'sed -i -e
"s/.*PasswordAuthentication.*/PasswordAuthentication yes/"
/etc/ssh/sshd_config' --run-command 'sed -i -e
"s/.*PermitRootLogin.*/PermitRootLogin yes/" /etc/ssh/sshd_config'
$ sudo systemctl stop libvirtd
```

**Upload the overcloud images to the Undercloud Glance repository**

```
$ openstack overcloud image upload --image-path /home/stack/images/
$ openstack image list
```

**Set the DNS Names servers on the undercloud subnet**

Get the UUID of the undercloud subnet by executing:

```
$ openstack subnet list
```

Update the DNS names servers by passing the UUID of the subnet and the DNS name servers

```
$ openstack subnet set --dns-nameservers 10.5.30.160 03dae028-f9bf-47dc-
bc2b-d54a72910abc
```

**Create instack.json and perform introspection**

The introspection process will contact each node to be used in the overcloud deployment and build an inventory that will be stored in the swift data of the undercloud. The first step in performing an introspection is to create a instackenv.json file that contains authentication and connection information for each node. This reference architecture was tested with both the generic pxe_impitool driver and the pxe_ilo. The pxe_ilo driver is documented in this reference architecture. A new local user was created in iLo named root with the following account privileges; Administer User Accounts, Remote Console Access, Virtual Power and Reset, Virtual Media, and Configure iLO Settings. This account was used to perform the introspection.

Below is an excerpt from the *instackenv.json* file. Refer to appendix B for an example of the full instackenv.json.

```
"nodes":[
     {
 "pm_type":"pxe_ilo",
 "mac":[
      "94:18:82:08:f0:14"
       ],
 "capabilities": "profile:ceph-storage,boot_option:local",
 "cpu":"2",
 "memory":"4096",
 "disk":"146",
 "arch":"x86_64",
 "pm_user":"root",
 "pm_password":"redhat",
 "pm_addr":"10.19.20.140"
    }
```

Notice the line "capabilities": "profile:ceph-storage,boot_option:local" in the above entry. This assigns the ceph-storage profile to the ProLiant HPE DL380 Gen 9 with the iLO address of 10.19.20.140. The ceph-storage profile is assigned to the each on the three HPE prolaint DL380 Gen9 servers that are configured with additional drives for Ceph OSD and journals. A similar entry must be created for each node where introspection will be performed. Refer to Appendix B for complete *instack.json* file example.

Import the instack.json file to register the node with Red Hat OpenStack Platform director using the following command:

```
$ openstack baremetal import --json ~/instackenv.json
$ openstack baremetal configure boot
```

**Verify the introspection pxe images are installed in /httpboot**

- agent.kernel
- agent.ramdisk
- inspector.ipxe

**Run the introspection:**

**Import the instackenv.json file**

```
$ openstack baremetal import --json ~/instackenv.json
```

**Configure the boot**

```
$ openstack baremetal configure boot
```

**List the imported baremetal nodes**
All nodes should show a Provisioning State of *available* and the Power State is *power off*.

```
$ openstack baremetal node list

| UUID | Name | Instance UUID | Power State | Provisioning State |
Maintenance
| af95182a-3107-423c-a9fa-8b14fb44d825 | None | None | power off
|available | False
| 93f85651-aba0-436b-95d4-783d81622960 | None | None | power off
|available | False
| 2faac1e8-715e-4f17-8314-7bace3e4ec01 | None | None | power off
|available | False
| 27f4be16-88dd-466f-a80c-cfaa3c8dec09 | None | None | power off
|available | False
| bbc764a0-e66b-406e-8c02-a2b2da21b38c | None | None | power off
|available | False
| 1138f2f3-ebfc-43c5-8ac0-e28a62f6be21 | None | None | power off
|available | False
| bd3c80f3-92e7-4e75-8272-8ad73bd7efed | None | None | power off
|available | False
| efeb181e-dd3a-4ad1-bc03-dbcfedb2bc97 | None | None | power off
|available | False
| 858f277f-5dd8-457d-9510-45d22173fc1e | None | None | power off
|available | False
| 87132020-612d-46b9-99ed-0a1509e67254 | None | None | power off
|available | False
```

**Set all nodes from available to manageable**

```
$ for node in $(openstack baremetal node list --fields uuid -f value) ;
do openstack baremetal node manage $node ; done
```

Execute *openstack baremetal node list* to verify the Provisioning State has been set to *manageable* as shown below.

```
$ openstack baremetal node list

| UUID | Name | Instance UUID | Power State | Provisioning State |
Maintenance
| af95182a-3107-423c-a9fa-8b14fb44d825 | None | None | power off
|manageable | False
| 93f85651-aba0-436b-95d4-783d81622960 | None | None | power off
|manageable | False
| 2faac1e8-715e-4f17-8314-7bace3e4ec01 | None | None | power off
|manageable | False
| 27f4be16-88dd-466f-a80c-cfaa3c8dec09 | None | None | power off
|manageable | False
| bbc764a0-e66b-406e-8c02-a2b2da21b38c | None | None | power off
|manageable | False
| 1138f2f3-ebfc-43c5-8ac0-e28a62f6be21 | None | None | power off
|manageable | False
| bd3c80f3-92e7-4e75-8272-8ad73bd7efed | None | None | power off
|manageable | False
| efeb181e-dd3a-4ad1-bc03-dbcfedb2bc97 | None | None | power off
|manageable | False
| 858f277f-5dd8-457d-9510-45d22173fc1e | None | None | power off
|manageable | False
| 87132020-612d-46b9-99ed-0a1509e67254 | None | None | power off
|manageable | False
```

**Perform the introspection**

```
$ openstack overcloud node introspect --all-manageable --provide
```

**Monitor the progress**

```
$ sudo journalctl -l -u openstack-ironic-inspector -u openstack-ironic-
inspector-dnsmasq -u openstack-ironic-conductor -f
```

When the introspection is complete the following message should be displayed on the screen

```
Started Mistral Workflow. Execution ID:
625f7c8e-adb0-4541-a9f1-a282dc4c562b
Waiting for introspection to finish...
Introspection for UUID 1138f2f3-ebfc-43c5-8ac0-e28a62f6be21 finished
successfully.
Introspection for UUID 93f85651-aba0-436b-95d4-783d81622960 finished
successfully.
Introspection for UUID efeb181e-dd3a-4ad1-bc03-dbcfedb2bc97 finished
successfully.
Introspection for UUID 87132020-612d-46b9-99ed-0a1509e67254 finished
successfully.
Introspection for UUID 858f277f-5dd8-457d-9510-45d22173fc1e finished
successfully.
Introspection for UUID af95182a-3107-423c-a9fa-8b14fb44d825 finished
successfully.
Introspection for UUID 27f4be16-88dd-466f-a80c-cfaa3c8dec09 finished
successfully.
Introspection for UUID 2faac1e8-715e-4f17-8314-7bace3e4ec01 finished
successfully.
```

```
Introspection for UUID bd3c80f3-92e7-4e75-8272-8ad73bd7efed finished
successfully.
Introspection for UUID bbc764a0-e66b-406e-8c02-a2b2da21b38c finished
successfully.
Introspection completed.
```

**List the Nodes after Introspection**

Once the introspection is complete executing the *openstack baremetal node list* command should show the Provisioning State as *available*, Maintenance as False, and Power State is power off for each of the baremetal nodes.

```
$ openstack baremetal node list

| UUID | Name | Instance UUID | Power State |*Provisioning State |
Maintenance
| 35b90609-6783-4958-b84f-a8415cd49438 | None | None | power off |
available | False
| 152cc51f-2dd5-474e-a04d-a029fac39175 | None | None | power off |
available | False
| 6761cb95-b5c6-44ad-931e-dd3bc1de92f9 | None | None | power off |
available | False
| d5ba620e-11ef-4ce3-bfce-ce60c927a5eb | None | None | power off |
available | False
| 138963ce-02f9-4944-90eb-16513c124727 | None | None | power off |
available | False
| df3453a9-c86a-4620-81ff-45254468860c | None | None | power off |
available | False
| eed0b799-9f35-472d-a43f-89442a1bc48b | None | None | power off |
available | False
| d1bd0e50-ff9d-4654-9830-b36b2223e914 | None | None | power off |
available | False
| 61826a3a-5381-44bd-b92d-2fd89da00b4d | None | None | power off |
available | False
| 24670e3b-e34e-4489-94d0-b9a0a1aa23ae | None | None | power off |
available | False
```

**List the Profiles**

Executing the *openstack overcloud profiles list* command will display the Node UUID, Provisioning State, and the Current Profile. The Current Profile was assigned in the *instackenv.json "capabilities"* setting for each node. No additional profile have been assigned and the nodes have not been deploy, so the Node Name and Possible Profiles columns are blank.

```
$ openstack overcloud profiles list

| Node UUID | Node Name | Provision State | Current Profile | Possible
Profiles
| af95182a-3107-423c-a9fa-8b14fb44d825 |    | available | compute |
|
| 93f85651-aba0-436b-95d4-783d81622960 |    | available | compute |
|
| 2faac1e8-715e-4f17-8314-7bace3e4ec01 |    | available | compute |
|
| 27f4be16-88dd-466f-a80c-cfaa3c8dec09 |    | available | compute |
|
```

```
| bbc764a0-e66b-406e-8c02-a2b2da21b38c |    | available | control |
|
| 1138f2f3-ebfc-43c5-8ac0-e28a62f6be21 |    | available | control |
|
| bd3c80f3-92e7-4e75-8272-8ad73bd7efed |    | available | control |
|
| efeb181e-dd3a-4ad1-bc03-dbcfedb2bc97 |    | available | ceph-storage |
|
| 858f277f-5dd8-457d-9510-45d22173fc1e |    | available | ceph-storage |
|
| 87132020-612d-46b9-99ed-0a1509e67254 |    | available | ceph-storage |
|
```

**Configure Red Hat Ceph Storage Nodes**

The next set of commands will review the introspection inventory for the Red Hat Ceph Storage nodes and set the root disks. The root disk should be set for all the Red Hat OpenStack Platform 10 nodes. Since the Controller and Compute nodes have a single logical disk, it is not as critical as setting the root disk for the Red Hat Ceph Storage nodes, which have multiple logical disks. To accomplish this the ironic introspection data for each node will be exported to a new directory under */home/stack*.

**Checking the ironic inventory for the disks**

**Export the ironic password:**

Create a directory for the exported data in the stack user's home directory */home/stack/*

```
$ mkdir swift-data
```

```
$ export SWIFT_PASSWORD=`sudo crudini --get /etc/ironic-
inspector/inspector.conf swift password`
```

SWIFT_PASSWORD=ff25ca2e11ebf0373f7854788c3298b8767688d7

The above command requires installing the crudini package on the Red Hat OpenStack Platform director. Alternatively, this password can be found in */home/stack/undercloud-passwords.conf*

undercloud_ironic_password=ff25ca2e11ebf0373f7854788c3298b8767688d7

Execute the following commands to export the introspection data for each node.

```
$ cd swift-data
$ for node in $(ironic node-list | grep -v UUID| awk '{print $2}'); do
swift -U service:ironic -K $SWIFT_PASSWORD download ironic-inspector
inspector_data-$node; done
```

This command will create a file for each node that contains the introspection data. Use the following command to identify the disks and device names for each node.

```
$ for node in $(ironic node-list | grep -v UUID| awk '{print $2}'); do
echo "NODE: $node" ; cat inspector_data-$node | jq '.inventory.disks' ;
echo "-----" ; done
```

This output will display the following:

```
NODE: af95182a-3107-423c-a9fa-8b14fb44d825
```

```
[
 {
  "size": 1200210141184,
  "rotational": true,
  "vendor": "HP",
  "name": "/dev/sda",
  "wwn_vendor_extension": "0x947935a19c341de5",
  "wwn_with_extension": "0x600508b1001cee69947935a19c341de5",
  "model": "LOGICAL VOLUME",
  "wwn": "0x600508b1001cee69",
  "serial": "600508b1001cee69947935a19c341de5"
 }
]
```

Refer back to the Profile List, generated by the openstack overcloud profile list command, comparing the NODE: af95182a-3107-423c-a9fa-8b14fb44d825 to the Node ID UUID in the profile list table, we can see that this data belongs to a node that has the compute profile assigned. The compute node and controller nodes will only display a single logical disk. The ceph-storage nodes will display 13 logical disks, one for the operating system, 12 for ceph osd, and two SSD drives for ceph journal files.

The “serial” key reflects the Drive Unique ID of the logical drives created using the HP Smart Storage Administrator. This information was captured earlier in the section titled “**Red Hat Ceph Storage Configuration”** when creating the logical drives on the Red Hat Ceph Storage nodes.

The journal SSD drives can be identified by the “rotational : false” key pair value.

In our example the device names for the Red Hat Ceph Storage nodes use the following conventions:

- /dev/sda is the operating system
- /dev/sdb - /dev/sdk are used for ceph osd
- /dev/sdl and /dev/sdm are solid state drives used for ceph journals.

The device names for the osd and journal files will be passed as parameters for the storage environment in the *extraParameters.yaml* file defined in the next section.

Perform the following commands to set the root disks on the ceph-storage nodes.

**Red Hat Ceph Storage node 1**

```
$ ironic node-update d294282a-3136-4324-9d92-0531432a94d6 add
properties/root_device='{"serial":"600508b1001c34622cdd92447070f364"}'
```

**Red Hat Ceph Storage node 2**

```
$ ironic node-update ae8d1b83-3387-4568-97d8-38e52279f422 add
properties/root_device='{"serial":"600508b1001c98c51329ff4030ab314f"}'
```

**Red Hat Ceph Storage node 3**

```
$ ironic node-update f96decbd-7057-470f-8575-293f4a1a2811 add
properties/root_device='{"serial":"600508b1001c19d84afd81f166812fd3"}'
```

**Red Hat Ceph Storage Tuning**

The following setting for the default journal size is set during the installation.

ceph::profile::params::osd_journal_size: 5120

Configuring the placement groups will be discussed in the next section.

**Heat Templates**

In the stack user's home directory create a directory structure to hold any customized Heat templates. In this reference architecture there six files that hold our customized configuration templates and scripts.
Those files include:

- templates/mytemplates/extraParams.yaml
- templates/mytemplates/nic-configs/compute.yaml
- templates/mytemplates/nic-configs/controller.yaml
- templates/mytemplates/nic-configs/ceph-storage.yaml
- templates/mytemplates/first-boot/wipe-disks.yaml
- templates/mytemplates/first-boot/wipe-disks.sh

The contents of these files can be found in Appendix B of this document. Updated versions can also be found at:
https://github.com/RHsyseng/OSP-HPE

Modifications to the network environment, storage, monitoring, and logging environments are consolidated into the extraParameters.yaml file. The nic-configs directory contains the network configuration files for the controller, compute, and ceph-storage nodes. These files provide the configuration information for creating the bond interfaces, assigning the bonded interfaces to their respective bridge, and assigning the VLANs to the interface for the storage and cloud management trunks.

The *wipe-disks.yaml* and the *wipe-disks.sh* will be executed on the first boot of the ceph nodes to wipe out the disks designated for OSD and journals.

**Network**

The ControlPlane network uses the eno1 interface on each of the servers, this interface is used for communication between Red Hat OpenStack Platform 10 director and the Red Hat OpenStack Platform 10 compute, controller, and Red Hat Ceph Storage nodes. The variables for the ControlPlaneIP, ControlPlaneSubnetCidr, and EC2MetadataIP are defined in the extraParameters.yaml file. Below is an excerpt from the *controller.yaml* file. Refer to Appendix B to view the entire *controller.yaml*, *compute.yaml*, and *ceph-storage.yaml* network configuration files.

```
type: interface
           name: eno1
           use_dhcp: false
           addresses:
             -
                ip_netmask:
                 list_join:
                   - '/'
                   - - {get_param: ControlPlaneIp}
                     - {get_param: ControlPlaneSubnetCidr}
```

```
            routes:
              -
                ip_netmask: 169.254.169.254/32
                next_hop: {get_param: EC2MetadataIp}
```

**Bonding**

The bonded interfaces are defined in the network configuration files, controller.yaml, compute.yaml, and ceph-storage.yaml. The cloud networking bond that contains the Internal API, External, Storage Management, and Tenant networks, are defined as an ovs_bond and attached to an ovs_bridge in the configuration files. The ovs_bridge is defined in the extraParameter.yaml file and the ovs_bond name is bond0. An excerpt of the *controller.yaml* file is shown below:

```
            -
              type: ovs_bridge
              name: {get_input: bridge_name}
              dns_servers: {get_param: DnsServers}
              members:
                -
                  type: ovs_bond
                  name: bond0
                  bonding_options: {get_param: BondInterfaceOvsOptions}
                  members:
                    -
                      type: interface
                      name: eno49
                      mtu: 9000
                      primary: true
                    -
                      type: interface
                      name: ens2f0
                      mtu: 9000
                -
                  type: vlan
                  device: bond0
                  vlan_id: {get_param: ExternalNetworkVlanID}
                  addresses:
                    -
                      ip_netmask: {get_param: ExternalIpSubnet}
                  routes:
                    -
                      default: true
                      next_hop: {get_param: ExternalInterfaceDefaultRoute}
                -
                  type: vlan
                  device: bond0
                  vlan_id: {get_param: InternalApiNetworkVlanID}
                  addresses:
                    -
                      ip_netmask: {get_param: InternalApiIpSubnet}
                -
                  type: vlan
                  device: bond0
                  vlan_id: {get_param: TenantNetworkVlanID}
                  addresses:
                    -
```

```
                ip_netmask: {get_param: TenantIpSubnet}
              -
               type: vlan
               device: bond0
               mtu: 9000
               vlan_id: {get_param: StorageMgmtNetworkVlanID}
               addresses:
                 -
                  ip_netmask: {get_param: StorageMgmtIpSubnet}
```

The Storage network is also defined in the network configuration files. In this case the ovs_bridge name is defined in the configuration file as br_storage and the ovs_bond name is bond1. Below is an excerpt from the *controller.yaml* file:

```
            -
             type: ovs_bridge
             name: br_storage
             dns_servers: {get_param: DnsServers}
             members:
               -
                type: ovs_bond
                name: bond1
                bonding_options: {get_param: BondInterfaceOvsOptions}
                members:
                  -
                   type: interface
                   name: eno50
                   primary: true
                   mtu: 9000
                  -
                   type: interface
                   name: ens2f1
                   mtu: 9000
               -
                type: vlan
                device: bond1
                mtu: 9000
                vlan_id: {get_param: StorageNetworkVlanID}
                addresses:
                  -
                   ip_netmask: {get_param: StorageIpSubnet}
```

The network interfaces used for these bonds are 10Gigabit interfaces. Bond0 uses the two 10Gb interfaces, eno49 and ens2f0. Bond1 uses the two 10Gigabit interfaces eno50 and ens2f1. The BondInterfaceOvsOptions for the ovs bonded interfaces is set to "slb-balance lacp=off" and defined in the *extraParameters.yaml* file. The vlan parameters, VlanID and IpSubnets, are also defined in the *extraParameters.yaml* file.

**Monitoring and Logging Configuration**

Monitoring is performed with a Sensu client that sends alerts to a preconfigured Sensu monitoring server and presented through a Uchiwa dashboard. Figure 3 depicts the Uchiwa dashboard to a Sensu server that is recieving alerts from the Red Hat Openstack Platform 10 deployment used in this reference architecture.

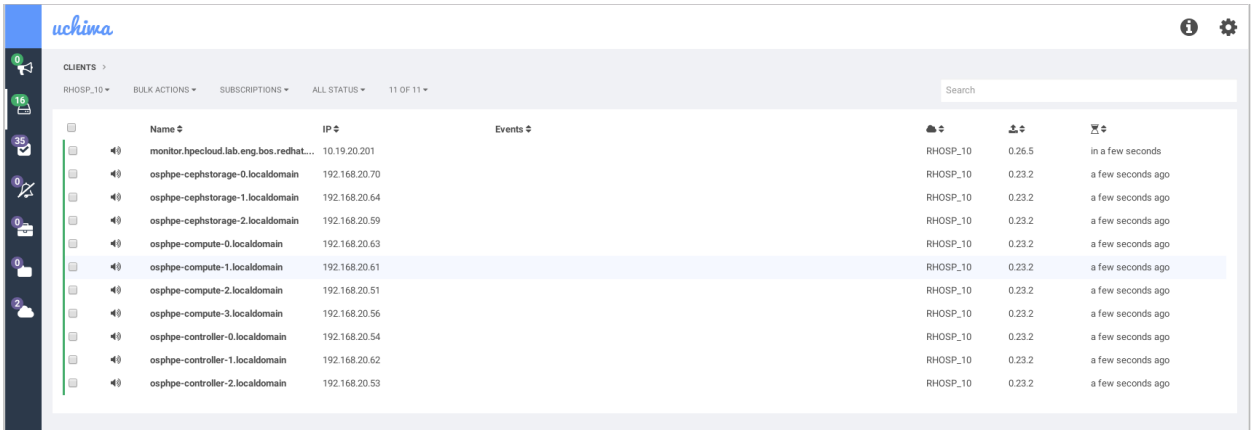



*Figure 3: Uchiwa dashboard for Sensu Monitoring*

The log files are collected using fluentd and sent to a preconfigured logging server, then presented via a Kibana dashboard. The Kibana dashboard shown in Figure 4 illustrates the event logging from the Red Hat Openstack Platform 10 deployment used in this reference architecture.

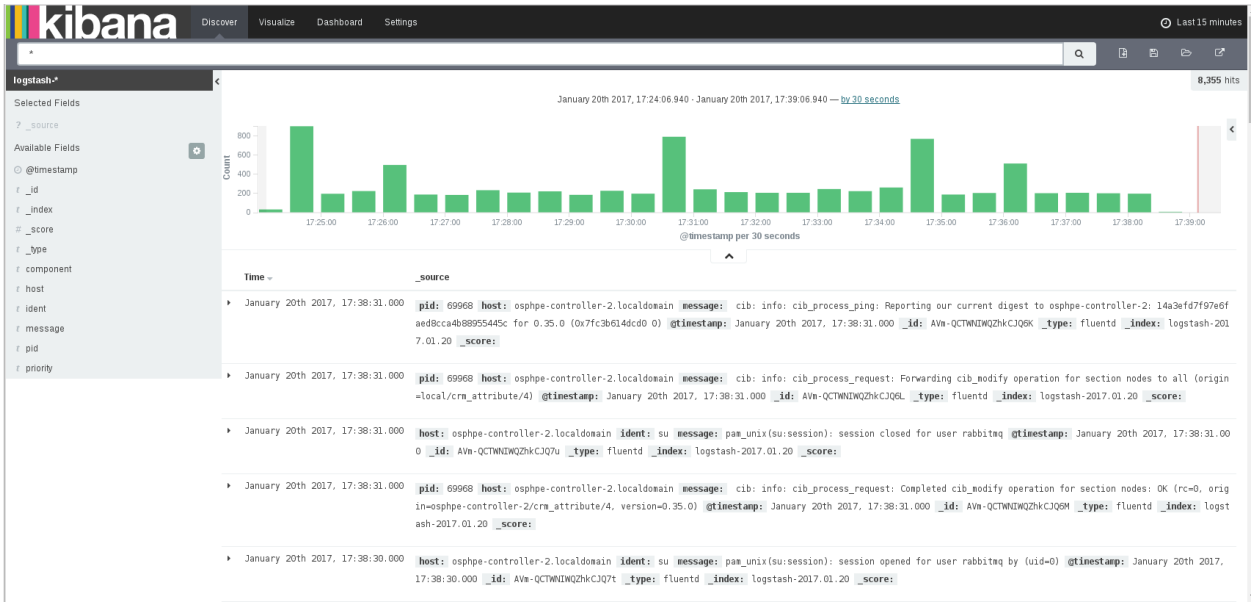



*Figure 4: Kibana dashboard for Fluentd Logging*

Including the monitoring and logging environments in the Red Hat OpenStack Platform 10 deployment will install the client side monitoring and logging components on the Red Hat OpenStack Platform 10 overcloud nodes. The server side components (Sensu, Fluentd, and Kibanna) for receiving monitors and log files are not installed as part of the Red Hat OpenStack Platform 10 deployment. For an automated deployment of the Sensu monitoring server and Fluentd logging server refer to the opstools-ansible git repository. The opstools-ansible project is an opensource community supported project and not part of this reference architecture. The ansible playbooks in this repository will install and configure the server side components to work with the monitoring and logging clients on the Red Hat OpenStack Platform 10 overcloud nodes. The opstools-ansible git project installs OpenStack specific Sensu checks required to monitor the OpenStack services. The Sensu client will automatically subscribe to these checks during the Red Hat OpenStack Platform 10 deployment. The monitoring and logging environments are configured by specifying the parameter variables in the *extraParameters.yaml*. Below is an excerpt for the *extraParameters.yaml* file that illustrates the monitoring and logging variables:

```
#Monitoring Parameters
 MonitoringRabbitHost: 192.168.20.201
 MonitoringRabbitPort: 5672
```

```
MonitoringRabbitUserName: sensu
MonitoringRabbitPassword: sensu
MonitoringRabbitUseSSL: false
MonitoringRabbitVhost: "/sensu"
#Logging
parameter_defaults:
LoggingServers:
  - host: 192.168.20.202
    port: 24224
```

In this example, the Sensu monitoring is installed on 192.168.20.201 and the Logging server is installed on 192.168.20.202. These servers must be available in the existing infrastructure or installed separately. The Sensu monitoring server and Fluentd logging server are not installed as part of the Red Hat OpenStack Platform 10 deployment or this reference architecture. The opstools-ansible project, found at https://github.com/centos-opstools/opstools-ansible, provides ansible playbooks that will install and configure the Sensu monitoring and Logging servers. In this example monitoring and logging is configured to communicate over the control plane network. The complete *extraParameters.yaml* file can be found in Appendix B at the end of this document.

## 5.2. DEPLOY THE OVERCLOUD

**Installing the Overcloud from a Command line**

The following script launches our openstack overcloud deploy command and will deploy an overcloud named osphpe:

```
source stackrc
openstack overcloud deploy \
--templates \
-e/usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e/usr/share/openstack-tripleo-heat-templates/environments/storage-
environment.yaml \
-e/usr/share/openstack-tripleo-heat-templates/environments/monitoring-
environment.yaml \
-e/usr/share/openstack-tripleo-heat-templates/environments/logging-
environment.yaml \
-e /home/stack/templates/mytemplates/extraParams.yaml \
--stack osphpe \
--debug \
--log-file overcloudDeploy.log \
--ceph-storage-flavor ceph-storage \
--ceph-storage-scale 3 \
--control-flavor control \
--control-scale 3 \
--compute-flavor compute \
--compute-scale 4 \
--block-storage-scale 0 \
--swift-storage-scale 0 \
--ntp-server 10.16.255.1 \
```

The --templates option in the *openstack overcloud deploy* command specifies the location of the environment templates that will be used during the deployment. The extraParams.yaml file is used to specify custom environment parameters (variables) for the network, storage, logging, and monitoring environments. The --stack option defines the name for the overcloud that will be

deployed.

**Installing the Overcloud with GUI**

Red Hat OpenStack Platform 10 supports the overcloud deployment using the TripleO GUI. This installation offers the installer the ability to customize the overcloud environment, validate the configuration, and launch the overcloud deployer.
Demonstrating the GUI installer is beyond the scope of this document.

**Monitoring the Deployment**

Below are some commands that are helpful in monitoring the overcloud deployment.

- openstack baremetal node list
- openstack server list
- openstack stack event list osphpe
- openstack stack resource list osphpe -n5

Additionally, opening an iLO remote console will allow the deployer to view the provisioning of the Red Hat OpenStack Platform 10 nodes. This may not be feasible for larger environments.
A successful completion the overcloud deploy script should return a 0 with following message that includes the Overcloud Endpoint URL:

```
Overcloud Endpoint: http://10.19.20.157:5000/v2.0
Overcloud Deployed
clean_up DeployOvercloud:
END return value: 0
```

The deployment should take approximately one hour to complete. If the deployment is taking considerably longer than one hour, it will eventually time out and return a 1, indicating an error condition. Refer to Appendix C for help with troubleshooting a failed deployment.

**Accessing the Overcloud**

Once the overcloud has been successfully deployed, an environment file for the overcloud is created in the stack user's home directory. The default file name is overcloudrc. The actual file name will be the name of the stack with rc appended to it. In this reference architecture the overcloud stack name is osphpe, therefore the environment file that is created as part of the Red Hat OpenStack Platform 10 deployment is */home/stack/osphperc*. This file contains the openstack environment variables for accessing the overcloud, including the default OS_USERNAME, OS_PASSWORD, OS_AUTH_URL, and OS_TENANT_NAME. Below is an example of the osphperc file:

```
export OS_NO_CACHE=True
export OS_CLOUDNAME=osphpe
export OS_AUTH_URL=http://10.19.20.154:5000/v2.0
export NOVA_VERSION=1.1
export COMPUTE_API_VERSION=1.1
export OS_USERNAME=admin
export
no_proxy=,10.19.20.156,192.168.20.58,10.19.20.158,192.168.20.58,10.19.20.
154,192.168.20.60
```

```
export OS_PASSWORD=4Gpy773FwnzGJsVevyP2EVbmT
export PYTHONWARNINGS="ignore:Certificate has no, ignore:A true
SSLContext object is not available"
export OS_TENANT_NAME=admin
```

Source */home/stack/osphperc* file to execute openstack commands against the newly deployed overcloud named osphpe. Additionally, the Red Hat OpenStack Platform 10 Horizon interface can be accessed from a browser using the OS_AUTH_URL IP Address. The user name and password for the horizon interface are also defined in the *osphperc* file as; OS_USERNAME=admin and OS_PASSWORD=4Gpy773FwnzGJsVevyP2EVbmT. The admin password, OS_PASSWORD, is randomly generated and will be unique for each Red Hat OpenStack Platform 10 deployment.

# CHAPTER 6. POST DEPLOYMENT AND VALIDATION

This section provides post deployment and validation information to ensure both the Red Hat OpenStack Platform 10 and the Red Hat Ceph Storage cluster are operational. After the successful deployment of the overcloud, verify the ceph cluster is operational and modify the configuration as necessary using the ceph commands described in this section. To ensure proper operation of the Red Hat OpenStack Platform 10, deploy a test virtual machine instance and configure a Floating IP Address to test external network connectivity.

## 6.1. RED HAT CEPH STORAGE POST DEPLOYMENT

**Verify the Red Hat Ceph Storage deployment**

The Red Hat Ceph Storage monitors are running on the overcloud controller nodes. The controller nodes are accessible using ssh and the heat-admin user account via the control plane IP address assigned to each controller node. To determine the control plane IP address for each of the Red Hat OpenStack Platform nodes, execute *openstack server list* using the stack user from the Red Hat OpenStack Platform director.

```
$ source stackrc
$ openstack server list
```

A table will be displayed showing the status of all the Red Hat OpenStack Platform nodes along with the control plane IP address. An execrpt from the output of the command is shown below. In our example the controller-0 control plane IP address is 192.168.20.54.

```
osphpe-controller-0  | ACTIVE | ctlplane=192.168.20.64 | overcloud-full
+
```

**Accessing and verifying the Red Hat Ceph Storage cluster**

From the Red Hat OpenStack Platform director, log into the overcloud controller using *ssh heat-admin@<controller-0 ctlplane IP address>*

The *ceph osd tree* command will display the individual OSD devices on each node.

```
$ sudo ceph osd tree
```

The output of the *ceph osd tree* command is shown below. This command will show the status of the individual OSD devices.

```
ID WEIGHT   TYPE NAME                     UP/DOWN REWEIGHT PRIMARY-
AFFINITY
-1 32.73285 root default
-2 10.91095     host osphpe-cephstorage-0
 0  1.09109         osd.0                       up  1.00000
1.00000
 1  1.09109         osd.1                       up  1.00000
1.00000
 4  1.09109         osd.4                       up  1.00000
1.00000
 5  1.09109         osd.5                       up  1.00000
1.00000
 8  1.09109         osd.8                       up  1.00000
```

```
1.00000
11  1.09109         osd.11                      up  1.00000
1.00000
13  1.09109         osd.13                      up  1.00000
1.00000
18  1.09109         osd.18                      up  1.00000
1.00000
21  1.09109         osd.21                      up  1.00000
1.00000
24  1.09109         osd.24                      up  1.00000
1.00000
-3 10.91095     host osphpe-cephstorage-2
 2  1.09109         osd.2                       up  1.00000
1.00000
 3  1.09109         osd.3                       up  1.00000
1.00000
 6  1.09109         osd.6                       up  1.00000
1.00000
 9  1.09109         osd.9                       up  1.00000
1.00000
12  1.09109         osd.12                      up  1.00000
1.00000
14  1.09109         osd.14                      up  1.00000
1.00000
16  1.09109         osd.16                      up  1.00000
1.00000
20  1.09109         osd.20                      up  1.00000
1.00000
23  1.09109         osd.23                      up  1.00000
1.00000
26  1.09109         osd.26                      up  1.00000
1.00000
-4 10.91095     host osphpe-cephstorage-1
 7  1.09109         osd.7                       up  1.00000
1.00000
10  1.09109         osd.10                      up  1.00000
1.00000
15  1.09109         osd.15                      up  1.00000
1.00000
17  1.09109         osd.17                      up  1.00000
1.00000
19  1.09109         osd.19                      up  1.00000
1.00000
22  1.09109         osd.22                      up  1.00000
1.00000
25  1.09109         osd.25                      up  1.00000
1.00000
27  1.09109         osd.27                      up  1.00000
1.00000
28  1.09109         osd.28                      up  1.00000
1.00000
29  1.09109         osd.29                      up  1.00000
1.00000
```

Launch *sudo ceph health* to verify ceph cluster health status.

```
$ sudo ceph health
HEALTH_WARN too few PGs per OSD (22 < min 30) +
```

The *ceph health* command will issue a warning due to insufficient placement groups. The default placement groups can be set as part the ceph variables in the *extraParams.yaml*. However, increasing the default value may not be appropriate for all OSD storage pools.

**Configure the number of ceph placement groups per pool**

List the ceph pools

```
$ sudo ceph osd lspools
0 rbd,1 metrics,2 images,3 backups,4 volumes,5 vms
```

Use the Ceph Placement Groups (PGs) per Pool Calculator to determine the correct number of placement groups for the OpenStack pools. The Ceph Placement Groups per Pool Calculator can be found at https://access.redhat.com/labs/cephpgc/. Below is an example of the recommendations generated from the ceph placement group calculator using OpenStack block storage with replicated ceph pools:

- backups 512
- volumes 512
- vms 256
- images 128
- metrics 128

**Increase Placement Groups**

The following commands can be used to increase the number of placement groups:

**ceph osd pool set {pool-name} pg_num {pg_num}**
**ceph osd pool set {pool-name} pgp_num {pgp_num}**

The rbd pool is not necessary and can be deleted. The following command will delete the rbd pool.

```
$ sudo ceph osd pool delete rbd rbd --yes-i-really-really-mean-it
```

For addtional post installation tuning refer to the Red Hat Ceph Storage documentation.

## 6.2. CREATE A TENANT AND DEPLOY AN INSTANCE

In this section we will perform the following operations to verify the operation of the Red Hat OpenStack Platform 10 cloud deployment:

- Create Tenant Project
- Create a Private Network
- Create and Configure a Subnet
- Create Router and Add the Router Interfaces
- Create Floating IPs

- Create Keypairs
- Create a Security Group
- Download and customize a cloud image
- Upload the cloud image to Glance
- Create Flavor for instance deployment
- Deploy an Instance

**Create an OpenStack Tenant**

The name of the OpenStack cloud in this example is osphpe, as mentioned in Chapter 5, an environment file was created in the stack user's home directory named *osphperc*. Source this file to ensure the openstack commands are executed using the overcloud (osphpe) keystone auth url and admin account.

```
$ source osphperc
```

Create a new OpenStack project called hpedemo-tenant.

```
$ openstack project create hpedemo-tenant
```

Create a new user account, this user account will be used for creating network resources and virtual machine instances in the new OpenStack project named hpedemo-tenant.

```
$ openstack user create hpeuser --password redhat
```

Add the hpeuser as a member to the hpedemo-tenant project.

```
$ openstack role add --user hpeuser --project hpedemo-tenant _member_
```

**Create an OpenStack environment file for the new OpenStack project**

Next, create an environment file named *keystonerc_hpedemo* (the name is not important). This file will set the environment to use the hpeuser account and the project to hpedemo-tenant.

```
$ vi keystonerc_hpedemo
[source]
export OS_USERNAME=hpeuser
export OS_TENANT_NAME=hpedemo-tenant
export OS_PASSWORD=redhat
export OS_CLOUDNAME=osphpe
export OS_AUTH_URL=<keystone auth URL>
$ source kestonerc_hpedemo
```

**Create a Private Network and Subnet**

Execute the following commands to create a private network for the hpedemo-tenant project.

```
$ openstack network list
$ openstack network create net1
$ openstack subnet create --name hpedemo-tenant-subnet net1 10.2.2.0/24
$ openstack subnet list
$ openstack subnet set --dns-nameserver <dns-IP Address> <subnet ID>
```

- 

**Create Router and assign interfaces**

Create a router and add an interface using the subnet ID from the *openstack subnet list* command executed in the previous step.

```
$ openstack router create router1
$ neutron router-interface-add router1 <subnet ID>
```

**Create the External Network and Allocate Floating IP Range**

The external network and subnet are created using the OpenStack admin credentials. Source the *osphperc* file to set the overcloud environment variables for the OpenStack admin user.

```
$ source osphperc
$ neutron net-create nova --router:external --provider:network_type vlan
--provider:physical_network datacentre --provider:segmentation 104
$ neutron subnet-create --name public --gateway 10.19.20.254 --
allocation-pool start=10.19.20.180,end=10.19.20.190 nova 10.19.20.128/25
```

Execute *openstack network list* to capture the UUID of the external network created above.

```
$ openstack network list
$ neutron router-gateway-set router1 <ExternalNet ID>
```

**Modify the Security Group to allow SSH and ICMP**

Modify the default security group for the hpedemo-tenant OpenStack project to allow icmp (ping) and ssh access to the virtual machine instances. Additionally, in the next section the virtual machine instances will be required to communicate with a Sensu monitoring host on port 5672 (rabbitmq). Open port 5672 to allow the virtual machine instances to communicate with a Sensu Monitoring host.

```
$ source keystonerc_hpedemo
$ openstack security group list
$ openstack security group rule create --ingress --protocol icmp <default
security group ID>
$ openstack security group rule create --ingress --dst-port 22 <default
security group ID>
$ openstack security group rule create --ingress --dst-port 5672 <default
security group ID>
$ openstack security group rule create --egress --dst-port 5672 <default
security group ID>
```

Verify the ports have been opened.

```
$ openstack security group show <security group ID>
```

**Create keypair**

Create a keypair in the hpedemo-tenant project and download the keypair to the local directory. The keypair will be used for ssh access by the default user (cloud-user) account in the Red Hat Enterprise Linux 7 cloud image.

```
$ openstack keypair create hpeuserkp > hpeuserkp.pem
```

The default permissions on the downloaded keypair file, *hpeuserkp.pem*, must be changed or a warning will be generated indicating that the private key file is unprotected and will be ignored. Change the keyfile permissions to read only for the user.

```
$ chmod 400 hpeuserkp.pem
```

**Download and customize a cloud image**

The Red Hat Enterprise Linux 7 cloud image will be used to test the ability to create virutal machine instances. The downloaded image is in qcow2 format. It is important to convert this virtual machine image format to raw.

```
$ sudo yum install -y rhel-guest-image-7.noarch
$ qemu-img convert -f qcow2 -O raw /usr/share/rhel-guest-image-7/*.qcow2
./rhel7.raw
```

The following command will customize the image to allow root login. Enabling root log in is an optional step which can be useful when accessing the virtual machine instance from the console and troubleshooting network connectivity issues.

```
$ sudo systemctl start libvirtd
$ virt-customize -a rhel7.raw --root-password password:redhat --run-
command 'sed -i -e "s/.*PasswordAuthentication.*/PasswordAuthentication
yes/" /etc/ssh/sshd_config' --run-command 'sed -i -e
"s/.*PermitRootLogin.*/PermitRootLogin yes/" /etc/ssh/sshd_config'
$ sudo systemctl stop libvirtd
```

**Upload the Image to Glance**

The guest image will now be uploaded into the Glance repository of the overcloud. Set the environment to use the hpeuser and hpedemo-tenant project (source *keystonerc_hpedemo*) to ensure the hpeuser created earlier has access to the image in the Glance repository.

```
$ source keystonerc_hpedemo
$ openstack image create --disk-format raw  --container-format bare --
file rhel7.raw rhel7
$ openstack image list
```

**Create a flavor for instance deployment**

In this version of OpenStack there are no default compute flavors. A new flavor must be created using the OpenStack admin account, set the environment to use the admin account (source *osphperc*).

```
$ source osphperc
$ openstack flavor create m1.small --id 1 --ram 4096 --disk 20 --vcpus
4
```

**Boot an new Virtual Machine Instance**

Create a virtual machine instance in the hpedemo-tenant project by executing the following commands.

```
$ source keystonerc_hpedemo
$ openstack server create --flavor 1 --image rhel7 --key-name hpeuserkp
inst1
```

```
$ openstack server list
```

The *openstack server list* command may have to be executed a few times until the virtual machine instance shows a state of *Active*, other states that are displayed prior to *Active* include *Spawning* and *Building*.

**Access the instance and verify connectivity**

The virtual machine instance must have a Floating IP address attached to the instance before the instance can be access from the External network. The following command will create a floating IP address and then attach the Floating IP address to the instance.

```
$ openstack floating ip create nova
$ openstack floating ip list
$ openstack server add floating ip inst1 <floating IP>
```

The virtual machine instance is now accessible from the Exernal network. Using the *ssh -i* command with the keypair that was previously downloaded and the Floating IP address, log into the virtual machine instance.

```
$ ssh -i hpeuserkp.pem cloud-user@<floating IP>
```

# CHAPTER 7. DEPLOY AN INSTANCE USING A HEAT TEMPLATE

In the previous section we deployed a single virtual machine instance to ensure basic operation of the Red Hat OpenStack Platform 10 cloud environment. In this section we will expand on this configuration by deploying a virtual machine instance using a heat template, registering the machine using Red Hat Subscription Manager, and then installing a Sensu monitoring client on the virtual machine instance.
Using a heat stack to create the virtual machine instance allows the user to create an instance and install software with a single command. The heat template used in this example can be found in Appendix B under the section titled "Heat Template for Instance Deployment"

In this example the heat template will be launched from the Red Hat OpenStack Platform director by the stack user. The template has been stored in a directory */home/user/stack/templates/mytemplates/inst/inst.yaml*. Additionally, source the environment file (*keystonerc_hpdemo*) to ensure the correct OpenStack username, password, and tenant environment variables are set.

```
$ source keystonerc_hpedemo
```

To launch the heat template execute the following command:

```
$ openstack stack create -t templates/mytemplates/inst/inst.yaml --
parameters "rhn_user=<rhnUserName>;rhn_password=<rhnPassword>;rhn_pool=
<rhnPoolID>;rhn_hostname=<instanceHostName>" <stack name>
```

In the command line example above, substitue the command line --parameter option variables with parameters that match your environment.

Use the *command _openstack stack list* to view stack that was created for the instance.

Execute *nova list* to view list virtual machine instances and obtain the virtual machine instance UUID for the virtul machine created by the heat stack template.

The command *openstack server show <instance UUID>* will provide details about the newly deployed instance.

```
$ openstack server show 6a4ca385-a901-48bb-8a51-eb301108ce13
```

The instance can be accessed by using ssh -i, *hpeuserkp.pem*, the cloud-user account, and the Floating IP address to log into the instance. This assumes the keypair (*hpeuserkp*) has been downloaded as *hpeuserkp.pem* in the current working directory.

```
$ ssh -i hpeuserkp.pem cloud-user@10.19.20.184
```

To delete the stack and the virtual machine instance use the command *openstack stack delete <stack name>*.

As described earlier, Red Hat OpenStack Platform 10 services are monitored by a Sensu server using Sensu subscriptions and checks. This monitoring functionality can be extended to virtual machine instances by adding a Sensu client to the virtual machine instances during deployment. The virtual machine instances can be registered with the existing Sensu server, or an additional Sensu server can be installed to provide a separate monitoring domain for the deployed services.

Both the virtual machine instance monitoring and OpenStack service monitoring can be viewed from a single instance of Uchiwa. Figure 5, shown below, illustrates the virtual machine instances registered with Sensu and displayed in the Uchiwa dashboard after being deployed using the heat template inst.yaml.

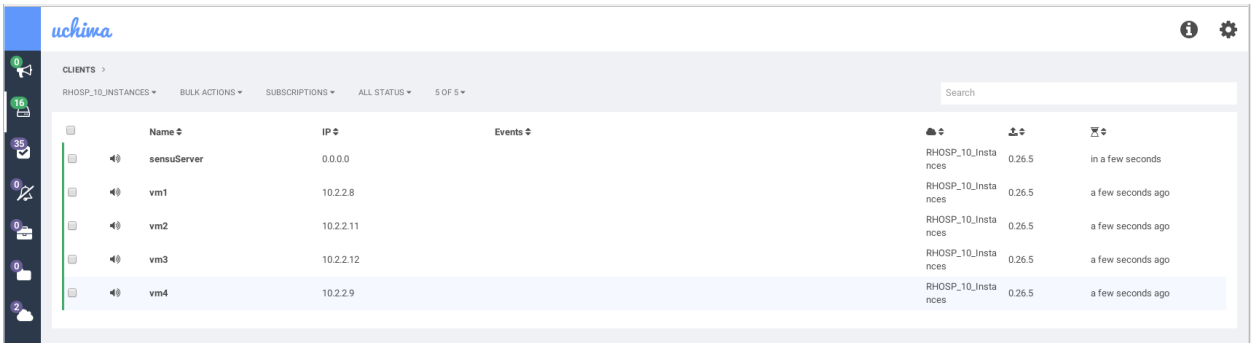



*Figure 5: Sensu Monitoring of Red Hat OpenStack Platform Instances*

To accomplish this we will include the installation of the Sensu client, erlang, and rabbitmq in the heat template that is used to deploy the instance.
The heat template will also create the Sensu client files necessary to register the instance with the existing Sensu server.
Below is an excerpt from the heat template that is used to create the instance.

```
            echo '{
               "client": {
                 "name": "localhost",
                 "address": "127.0.0.1",
                 "subscriptions": [
                   "test"
                 ]
               }
             }' | tee /etc/sensu/conf.d/client.json
             echo '{
              "rabbitmq": {
                "port": 5672,
                "host": "10.19.20.200",
                "user": "sensu",
                "password": "sensu",
                "vhost": "/sensu",
                "reconnect_on_error": false,
                "prefetch": 1
              }
            }' | tee /etc/sensu/conf.d/rabbitmq.json
              echo  '{
              "transport": {
                "name": "rabbitmq",
                "reconnect_on_error": true
              }
            }' | tee /etc/sensu/conf.d/transport.json
```

An example of the complete heat template can be found in Appendix B of this document.

The next step is to apply the instance hostname and IP Address to the Sensu *client.json* configuration file. A simple ansible script will be used to replace the "name: "localhost" and "address":"127.0.0.1" values in */etc/sensu/conf.d/client.json* file. This script will also start the sensu-client service on the instance. The ansible script can be pulled down from a git repository using the

heat template.

```
---
- hosts: all
  tasks:
     - name: replace ip
       replace:
         dest: /etc/sensu/conf.d/client.json
         regexp: "127.0.0.1"
         replace: "{{ ansible_eth0.ipv4.address }}"
         backup: no
     - name: replace hostname
       replace:
         dest: /etc/sensu/conf.d/client.json
         regexp: "localhost"
         replace: "{{ ansible_fqdn }}"
         backup: no
     - name: start Sensu Client
       shell: systemctl start sensu-client
```

The above script will register the Sensu client running on the instance with a Sensu server that is listening on 10.19.20.200:5672. The virtual machine instance should now appear in the Uchiwa dashboard. Additional application and operating system checks can be added to the Sensu server and subscribed to by the Sensu client running on the virtual machine instance.

# CHAPTER 8. CONCLUSION

This reference architecture provides installation and configuration details for a Red Hat OpenStack Platform 10 deployment on HPE ProLiant DL servers and HPE FlexFabric network switches. The reference architecture is comprised of 11 HPE ProLiant DL servers, one HPE ProLiant DL360 Gen9 server to host the Red Hat OpenStack Platform 10 director, three HPE ProLiant DL360 Gen9 server to host the Red Hat OpenStack Platform 10 controller nodes, four HPE ProLiant DL360 Gen9 servers to host the Red Hat OpenStack Platform 10 compute nodes, and three HPE ProLiant DL380 Gen9 servers to host the Red Hat Ceph Storage nodes. Detailed instructions were presented describing how to setup and configure the HPE ProLiant DL servers and HPE FlexFabric network switches. The deployment and configuration processes described in this reference architecture guides the installer or architect from baremetal servers and unconfigured switches to a fully functional Red Hat OpenStack Platform 10 deployment that is production ready; capable of deploying virtual machine instances through the Horizon GUI or from a commandline interface. The use of heat templates for deploying virtual machine instances and installing applications is also described as part of a use case scenario presented in this reference architecture. This solution provides increased availability through redundant Red Hat OpenStack Platform 10 server roles, Red Hat Ceph Storage clustering, and redundant hardware components including; redundant powersupplies, fans, and hard disk drives. Network redundancy is implemented through the use of bonded interfaces.

# APPENDIX A. BILL OF MATERIALS

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| **Director** | **DL360p Gen9 8SFF** | **AIOP-13024** | **3** |
| ProLiant DL360 Gen9 | HPE ProLiant DL360 Gen9 8SFF Configure-to-order ServerHPE Dynamic Smart Array B140i 8-Bay SFF Drive Cage | 755258-B21 | 1 |
| Processor 1 | Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W) FIO Processor Kit | 780003-L21 | 1 |
| Processor 2 | Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W) FIO Processor Kit | 780003-B21 | 1 |
| Memory for 1st processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| Memory for 2nd processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| FlexibleLOM | HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter | 665243-B21 | 1 |
| ALOM | 2 port 10Gb NIC | 727055-B21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| Power Supply | HPE 800W Flex Slot Platinum Hot Plug Power Supply Kit | 720479-B21 | 2 |
| Hard Drives | HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) SC Dual Port Enterprise 3yr Warranty Hard Drive | 718162-B21 | 2 |
| Storage Controller | HPE Smart Array P440ar/2GB FBWC 12Gb 2-ports Int FIO SAS Controller | 749974-B21 | 1 |
| Rail Kit | HPE 1U SFF Ball Bearing Rail Kit | 663201-B21 | 1 |
| SATA Cable | HPE DL360 SFF Embed SATA Cable | 766207-B21 | 1 |
| | | | |
| **Controllers** | **DL360p Gen9 8SFF** | **AIOP-13024** | **3** |
| ProLiant DL360 Gen9 | HPE ProLiant DL360 Gen9 8SFF Configure-to-order Server HPE Dynamic Smart Array B140i 8-Bay SFF Drive Cage | 755258-B21 | 1 |
| Processor 1 | Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W) FIO Processor Kit | 780003-L21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| Processor 2 | Intel Xeon E5-2699v3 (2.6GHz/18-core/45MB/145W) FIO Processor Kit | 780003-B21 | 1 |
| Memory for 1st processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| Memory for 2nd processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| FlexibleLOM | HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter | 665243-B21 | 1 |
| ALOM | 2 port 10Gb NIC | 727055-B21 | 1 |
| Power Supply | HPE 800W Flex Slot Platinum Hot Plug Power Supply Kit | 720479-B21 | 2 |
| Hard Drives | HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) SC Dual Port Enterprise 3yr Warranty Hard Drive | 718162-B21 | 2 |
| Storage Controller | HPE Smart Array P440ar/2GB FBWC 12Gb 2-ports Int FIO SAS Controller | 749974-B21 | 1 |
| Rail Kit | HPE 1U SFF Ball Bearing Rail Kit | 663201-B21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| SATA Cable | HPE DL360 SFF Embed SATA Cable | 766207-B21 | 1 |
| | | | |
| **Compute** | **DL360p Gen9 8SFF** | **AIOP-13024** | **4** |
| ProLiant DL360 Gen9 | HPE ProLiant DL360 Gen9 8SFF Configure-to-order Server HPE Dynamic Smart Array B140i 8-Bay SFF Drive Cage | 755258-B21 | 1 |
| Processor 1 | Intel Xeon E5-2690v3 (2.6GHz/12-core/30MB/135W) Processor | 755396-B21 | 1 |
| Processor 2 | Intel Xeon E5-2690v3 (2.6GHz/12-core/30MB/135W) FIO | 755396-L21 | 1 |
| Memory for 1st processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| Memory for 2nd processor | 16GB (1x16GB) Dual Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726719-B21 | 2 |
| FlexibleLOM | HPE Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter | 665243-B21 | 1 |
| ALOM | 2 port 10Gb NIC | 727055-B21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| Power Supply | HPE 800W Flex Slot Platinum Hot Plug Power Supply Kit | 720479-B21 | 2 |
| Hard Drives | HPE 1.2TB 6G SAS 10K rpm SFF (2.5-inch) SC Dual Port Enterprise 3yr Warranty Hard Drive | 718162-B21 | 2 |
| Storage Controller | HPE Smart Array P440ar/2GB FBWC 12Gb 2-ports Int FIO SAS Controller | 749974-B21 | 1 |
| Rail Kit | HPE 1U SFF Ball Bearing Rail Kit | 663201-B21 | 1 |
| SATA Cable | HPE DL360 SFF Embed SATA Cable | 766207-B21 | 1 |
| | | | |
| **Red Hat Ceph-Storage Nodes** | **DL380 Gen9 24SFF** | | **3** |
| ProLiant DL380 Gen9 | HP ProLiant DL380 Gen9 24SFF Configure-to-order Server HP Dynamic Smart Array B140i 24-Bay SFF Drive Cage | 767032-B21 | 1 |
| Storage | DL380 Gen9 2SFF Front/Rear SAS/SATA Kit | 724864-B21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| Processor | Intel Xeon E5-2643v3 (3.4GHz/6-core/20MB/135W) FIO Processor Kit | 719057-L21 | 1 |
| Memory | 8GB (1x8GB) Single Rank x4 DDR4-2133 CAS-15-15-15 Registered Memory Kit | 726718-B21 | 8 |
| FlexibleLOM | HP Ethernet 10Gb 2-port 560FLR-SFP+ FIO Adapter | 665243-B21 | 1 |
| ALOM | 2 port 10Gb NIC | 727055-B21 | 1 |
| Power Supply | HP 800W Flex Slot Platinum Hot Plug Power Supply Kit | 720479-B21 | 2 |
| Hard Drives | HP 1.2TB 6G SAS 10K rpm SFF (2.5-inch) SC Dual Port Enterprise 3yr Warranty Hard Drive | 718162-B21 | 2 |
| SSD Hard Drives | HP 400GB 12G SAS Write Intensive SFF 2.5-in SC 3yr Wty Solid State Drive | 802582-B21 | 1 |
| Storage Controller | HP Smart Array P440ar/2GB FBWC 12Gb 2-ports Int FIO SAS Controller | 749974-B21 | 1 |
| SAS Expander | HP 12G DL380 Gen9 SAS Expander Card | 727250-B21 | 1 |

| Name | Description | Part Number | Quantity |
|---|---|---|---|
| Rail Kit | HP 2U SFF Ball Bearing Rail Kit | 720863-B21 | 1 |
| | | | |
| **Rack Infrastructure and Networking** | | | |
| PDU | NA/JPN - HP 4.9kVA 208V 20out NA/JP bPDU | H5M58A | 4 |
| OpenStack Mgmt Switch | HP FF 5930 32QSFP+ Switch | JG726A | 1 |
| IPMI/iLO Switch | HP FF 5700-48G-4XG-2QSFP+ Switch | JG894A | 1 |
| Switch power supplies | HP A58x0AF 650W AC Power Supply | JC680A | 2 |
| Switch power supplies | HP A58x0AF 300W AC Power Supply | JG900A | 2 |
| Switch fan trays | HP X712 Bck(pwr)-Frt(prt) HV Fan Tray | JG553A | 2 |
| Switch fan trays | HP A58 X0Af Bck(pwr)-Frt(prt) Fan Tray | JC682A | 2 |
| Server split-out cables | HP X240 QSFP+ 4x10G SFP+ 5m DAC Cable | JG331A | 12 |
| 40Gb DAC cable for IRF | HP X240 40G QSFP+ QSFP+ 5m DAC Cable | JG328A | 1 |

*Table 10: Bill of Materials*

# APPENDIX B. TEMPLATES AND SCRIPTS

## B.1. OSPHPE-DEPLOY-OVERCLOUD.SH

```
#!/bin/bash -x
date
source stackrc
openstack overcloud deploy \
--templates \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-
isolation.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/storage-
environment.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/monitoring-
environment.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/logging-
environment.yaml \
-e /home/stack/templates/mytemplates/extraParams.yaml \
--stack osphpe \
--debug \
--log-file overcloudDeploy.log \
--ceph-storage-flavor ceph-storage \
--ceph-storage-scale 3 \
--control-flavor control \
--control-scale 3 \
--compute-flavor compute \
--compute-scale 4 \
--block-storage-scale 0 \
--swift-storage-scale 0 \
--ntp-server 10.16.255.1
date
```

## B.2. WIPE-DISKS.YAML

```
heat_template_version: 2014-10-16
description: >
  Wipe and convert all disks to GPT (except the disk containing the root
file system)
resources:
  userdata:
    type: OS::Heat::MultipartMime
    properties:
      parts:
      - config: {get_resource: wipe_disk}
  wipe_disk:
    type: OS::Heat::SoftwareConfig
    properties:
      config: {get_file: wipe-disk.sh}
outputs:
  OS::stack_id:
    value: {get_resource: userdataa}
```

## B.3. WIPE-DISK.SH

```
#!/bin/bash
if [[ `hostname` = *"ceph"* ]]
then
  echo "Number of disks detected: $(lsblk -no NAME,TYPE,MOUNTPOINT | grep
"disk" | awk '{print $1}' | wc -l)"
  for DEVICE in `lsblk -no NAME,TYPE,MOUNTPOINT | grep "disk" | awk
'{print $1}'`
  do
    ROOTFOUND=0
    echo "Checking /dev/$DEVICE..."
    echo "Number of partitions on /dev/$DEVICE: $(expr $(lsblk -n
/dev/$DEVICE | awk '{print $7}' | wc -l) - 1)"
    for MOUNTS in `lsblk -n /dev/$DEVICE | awk '{print $7}'`
    do
      if [ "$MOUNTS" = "/" ]
      then
        ROOTFOUND=1
      fi
    done
    if [ $ROOTFOUND = 0 ]
    then
      echo "Root not found in /dev/${DEVICE}"
      echo "Wiping disk /dev/${DEVICE}"
      sgdisk -Z /dev/${DEVICE}
      sgdisk -g /dev/${DEVICE}
    else
      echo "Root found in /dev/${DEVICE}"
    fi
  done
fi
```

## B.4. UNDERCLOUD.CONF

```
undercloud_hostname = director.hpecloud.lab.eng.bos.redhat.com
local_ip = 192.168.20.20/24
network_gateway = 192.168.20.20
undercloud_public_vip = 192.168.20.2
undercloud_admin_vip = 192.168.20.3
undercloud_service_certificate = /etc/pki/instack-certs/undercloud.pem
local_interface = ens8
network_cidr = 192.168.20.0/24
masquerade_network = 192.168.20.0/24
dhcp_start = 192.168.20.50
dhcp_end = 192.168.20.99
inspection_interface = br-ctlplane
inspection_iprange = 192.168.20.100,192.168.20.120
inspection_runbench = false
undercloud_debug = true
enable_mistral = true
```

```
enable_zaqar = true
enable_ui = true
enable_validations = true
ipxe_deploy = true
```

## B.5. EXTRAPARAMETER.YAML

```
resource_registry:
   OS::TripleO::Compute::Net::SoftwareConfig:
/home/stack/templates/mytemplates/nic-configs/compute.yaml
   OS::TripleO::Controller::Net::SoftwareConfig:
/home/stack/templates/mytemplates/nic-configs/controller.yaml
   OS::TripleO::CephStorage::Net::SoftwareConfig:
/home/stack/templates/mytemplates/nic-configs/ceph-storage.yaml
   OS::TripleO::NodeUserData:
/home/stack/templates/mytemplates/firstboot/wipe-disks.yaml
parameter_defaults:
  ControlPlaneSubnetCidr: "24"
  ControlPlaneDefaultRoute: 192.168.20.1
  EC2MetadataIp: 192.168.20.30
  DnsServers: ['10.11.5.19','8.8.8.8']
  # Customize all these values to match the local environment
  InternalApiNetCidr: 172.16.10.0/24
  StorageNetCidr: 172.16.6.0/24
  StorageMgmtNetCidr: 172.16.16.0/24
  TenantNetCidr: 172.16.160.0/24
  ExternalNetCidr: 10.19.20.128/25
  InternalApiAllocationPools: [{'start': '172.16.10.100', 'end':
'172.16.10.200'}]
  StorageAllocationPools: [{'start': '172.16.6.10', 'end':
'172.16.6.200'}]
  StorageMgmtAllocationPools: [{'start': '172.16.16.10', 'end':
'172.16.16.200'}]
  TenantAllocationPools: [{'start': '172.16.160.10', 'end':
'172.16.160.200'}]
  # Use an External allocation pool which will leave room for floating
IPs
  ExternalAllocationPools: [{'start': '10.19.20.150', 'end':
'10.19.20.175'}]
  InternalApiNetworkVlanID: 3041
  StorageNetworkVlanID: 3042
  StorageMgmtNetworkVlanID: 3043
  TenantNetworkVlanID: 3044
  ExternalNetworkVlanID: 104
  # Set to the router gateway on the external network
  ExternalInterfaceDefaultRoute: 10.19.20.254
  # Customize bonding options if required (will be ignored if bonds are
not used)
  BondInterfaceOvsOptions: 'balance-slb lacp=off'
  NeutronExternalNetworkBridge: "''"
  NeutronBridgeMappings: "datacentre:br-ex"
  NeutronNetworkVLANRanges: 'datacentre:1:1000'
  NeutronTunnelTypes: 'vxlan'
  TimeZone: 'US/Eastern'
  #Monitoring Parameters
```

```
  MonitoringRabbitHost: 192.168.20.201
  MonitoringRabbitPort: 5672
  MonitoringRabbitUserName: sensu
  MonitoringRabbitPassword: sensu
  MonitoringRabbitUseSSL: false
  MonitoringRabbitVhost: "/sensu"
  #Logging
  LoggingServers:
    - host: 192.168.20.202
      port: 24224
  parameter_defaults:
  ExtraConfig:
    ceph::profile::params::osd_journal_size: 5120
    ceph::profile::params::osds:
        '/dev/sdb':
          journal: '/dev/sdl1'
        '/dev/sdc':
          journal: '/dev/sdm1'
        '/dev/sdd':
          journal: '/dev/sdl2'
        '/dev/sde':
          journal: '/dev/sdm2'
        '/dev/sdf':
          journal: '/dev/sdl3'
        '/dev/sdg':
          journal: '/dev/sdm3'
        '/dev/sdh':
          journal: '/dev/sdl4'
        '/dev/sdi':
          journal: '/dev/sdm4'
        '/dev/sdj':
          journal: '/dev/sdl5'
        '/dev/sdk':
          journal: '/dev/sdm5'
```

## B.6. COMPUTE.YAML

```
heat_template_version: 2015-04-30
description: >
  Software Config to drive os-net-config with 2 bonded nics on a bridge
  with VLANs attached for the compute role.
parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
```

```
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  BondInterfaceOvsOptions:
    default: ''
    description: The ovs_options string for the bond interface. Set
things like
                 lacp=active and/or bond_mode=balance-slb using this
option.
    type: string
  InternalApiNetworkVlanID:
    default: 0
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 3042
    description: Vlan ID for the storage network traffic.
    type: number
  TenantNetworkVlanID:
    default: 0
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 0
    description: Vlan ID for the management network traffic.
    type: number
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
resources:
  OsNetConfigImpl:
```

```
type: OS::Heat::StructuredConfig
properties:
  group: os-apply-config
  config:
    os_net_config:
      network_config:
        -
          type: interface
          name: eno1
          use_dhcp: false
          dns_servers: {get_param: DnsServers}
          addresses:
            -
              ip_netmask:
                list_join:
                  - '/'
                  - - {get_param: ControlPlaneIp}
                    - {get_param: ControlPlaneSubnetCidr}
          routes:
            -
              ip_netmask: 169.254.169.254/32
              next_hop: {get_param: EC2MetadataIp}
            -
              default: true
              next_hop: {get_param: ControlPlaneDefaultRoute}
        -
          type: ovs_bridge
          name: {get_input: bridge_name}
          dns_servers: {get_param: DnsServers}
          members:
            -
              type: ovs_bond
              name: bond0
              bonding_options: {get_param: BondInterfaceOvsOptions}
              members:
                -
                  type: interface
                  name: eno49
                  primary: true
                -
                  type: interface
                  name: ens2f0
            -
              type: vlan
              device: bond0
              vlan_id: {get_param: InternalApiNetworkVlanID}
              addresses:
                -
                 ip_netmask: {get_param: InternalApiIpSubnet}
            -
              type: vlan
              device: bond0
              vlan_id: {get_param: TenantNetworkVlanID}
              addresses:
                -
                 ip_netmask: {get_param: TenantIpSubnet}
```

```
            -
              type: ovs_bridge
              name: br_storage
              dns_servers: {get_param: DnsServers}
              members:
                -
                  type: ovs_bond
                  name: bond1
                  bonding_options: {get_param: BondInterfaceOvsOptions}
                  members:
                    -
                      type: interface
                      name: eno50
                      primary: true
                      mtu: 9000
                    -
                      type: interface
                      name: ens2f1
                      mtu: 9000
                -
                  type: vlan
                  mtu: 9000
                  device: bond1
                  vlan_id: {get_param: StorageNetworkVlanID}
                  addresses:
                    -
                     ip_netmask: {get_param: StorageIpSubnet}
            -
              type: interface
              name: eno2
              use_dhcp: false
              defroute: false
            -
              type: interface
              name: eno3
              use_dhcp: false
              defroute: false
            -
              type: interface
              name: eno4
              use_dhcp: false
              defroute: false
outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

## B.7. CONTROLLER.YAML

```
heat_template_version: 2015-04-30
description: >
  Software Config to drive os-net-config with 2 bonded nics on a bridge
  with VLANs attached for the controller role.
parameters:
  ControlPlaneIp:
```

```
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  BondInterfaceOvsOptions:
    default: ''
    description: The ovs_options string for the bond interface. Set
things like
                 lacp=active and/or bond_mode=balance-slb using this
option.
    type: string
  ExternalNetworkVlanID:
    default: 0
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: 0
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 0
    description: Vlan ID for the storage network traffic.
    type: number
  StorageMgmtNetworkVlanID:
    default: 3042
    description: Vlan ID for the storage mgmt network traffic.
    type: number
  TenantNetworkVlanID:
    default: 0
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
```

```
    default: 0
    description: Vlan ID for the management network traffic.
    type: number
  ExternalInterfaceDefaultRoute:
    default: '10.0.0.1'
    description: default route for the external network
    type: string
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
resources:
  OsNetConfigImpl:
    type: OS::Heat::StructuredConfig
    properties:
      group: os-apply-config
      config:
        os_net_config:
          network_config:
            -
              type: interface
              name: eno1
              use_dhcp: false
              addresses:
                -
                  ip_netmask:
                    list_join:
                      - '/'
                      - - {get_param: ControlPlaneIp}
                        - {get_param: ControlPlaneSubnetCidr}
              routes:
                -
                  ip_netmask: 169.254.169.254/32
                  next_hop: {get_param: EC2MetadataIp}
            -
              type: ovs_bridge
              name: {get_input: bridge_name}
              dns_servers: {get_param: DnsServers}
              members:
                -
                  type: ovs_bond
                  name: bond0
                  bonding_options: {get_param: BondInterfaceOvsOptions}
                  members:
                    -
                      type: interface
                      name: eno49
                      mtu: 9000
```

```
                primary: true
              -
                type: interface
                name: ens2f0
                mtu: 9000
            -
              type: vlan
              device: bond0
              vlan_id: {get_param: ExternalNetworkVlanID}
              addresses:
                -
                  ip_netmask: {get_param: ExternalIpSubnet}
              routes:
                -
                  default: true
                  next_hop: {get_param: ExternalInterfaceDefaultRoute}
            -
              type: vlan
              device: bond0
              vlan_id: {get_param: InternalApiNetworkVlanID}
              addresses:
                -
                  ip_netmask: {get_param: InternalApiIpSubnet}
            -
              type: vlan
              device: bond0
              vlan_id: {get_param: TenantNetworkVlanID}
              addresses:
                -
                  ip_netmask: {get_param: TenantIpSubnet}
            -
              type: vlan
              device: bond0
              mtu: 9000
              vlan_id: {get_param: StorageMgmtNetworkVlanID}
              addresses:
                -
                  ip_netmask: {get_param: StorageMgmtIpSubnet}
        -
          type: ovs_bridge
          name: br_storage
          dns_servers: {get_param: DnsServers}
          members:
            -
              type: ovs_bond
              name: bond1
              bonding_options: {get_param: BondInterfaceOvsOptions}
              members:
                -
                  type: interface
                  name: eno50
                  primary: true
                  mtu: 9000
                -
                  type: interface
                  name: ens2f1
```

```
                    mtu: 9000
                -
                 type: vlan
                 device: bond1
                 mtu: 9000
                 vlan_id: {get_param: StorageNetworkVlanID}
                 addresses:
                   -
                    ip_netmask: {get_param: StorageIpSubnet}
              -
               type: interface
               name: eno2
               use_dhcp: false
               defroute: false
              -
               type: interface
               name: eno3
               use_dhcp: false
               defroute: false
              -
               type: interface
               name: eno4
               use_dhcp: false
               defroute: false
outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

## B.8. CEPH-STORAGE.YAML

```
heat_template_version: 2015-04-30
description: >
  Software Config to drive os-net-config with 2 bonded nics on a bridge
  with VLANs attached for the ceph storage role.
parameters:
  ControlPlaneIp:
    default: ''
    description: IP address/subnet on the ctlplane network
    type: string
  ExternalIpSubnet:
    default: ''
    description: IP address/subnet on the external network
    type: string
  InternalApiIpSubnet:
    default: ''
    description: IP address/subnet on the internal API network
    type: string
  StorageIpSubnet:
    default: ''
    description: IP address/subnet on the storage network
    type: string
  StorageMgmtIpSubnet:
    default: ''
    description: IP address/subnet on the storage mgmt network
```

```
    type: string
  TenantIpSubnet:
    default: ''
    description: IP address/subnet on the tenant network
    type: string
  ManagementIpSubnet: # Only populated when including
environments/network-management.yaml
    default: ''
    description: IP address/subnet on the management network
    type: string
  BondInterfaceOvsOptions:
    default: ''
    description: The ovs_options string for the bond interface. Set
things like
                 lacp=active and/or bond_mode=balance-slb using this
option.
    type: string
    constraints:
      - allowed_pattern: "^((?!balance.tcp).)*$"
        description: |
          The balance-tcp bond mode is known to cause packet loss and
          should not be used in BondInterfaceOvsOptions.
  ExternalNetworkVlanID:
    default: 0
    description: Vlan ID for the external network traffic.
    type: number
  InternalApiNetworkVlanID:
    default: 0
    description: Vlan ID for the internal_api network traffic.
    type: number
  StorageNetworkVlanID:
    default: 3042
    description: Vlan ID for the storage network traffic.
    type: number
  StorageMgmtNetworkVlanID:
    default: 0
    description: Vlan ID for the storage mgmt network traffic.
    type: number
  TenantNetworkVlanID:
    default: 0
    description: Vlan ID for the tenant network traffic.
    type: number
  ManagementNetworkVlanID:
    default: 0
    description: Vlan ID for the management network traffic.
    type: number
  ControlPlaneSubnetCidr: # Override this via parameter_defaults
    default: '24'
    description: The subnet CIDR of the control plane network.
    type: string
  ControlPlaneDefaultRoute: # Override this via parameter_defaults
    description: The default route of the control plane network.
    type: string
  ExternalInterfaceDefaultRoute: # Not used by default in this template
    default: '10.0.0.1'
    description: The default route of the external network.
```

```
    type: string
  ManagementInterfaceDefaultRoute: # Commented out by default in this
template
    default: unset
    description: The default route of the management network.
    type: string
  DnsServers: # Override this via parameter_defaults
    default: []
    description: A list of DNS servers (2 max for some implementations)
that will be added to resolv.conf.
    type: comma_delimited_list
  EC2MetadataIp: # Override this via parameter_defaults
    description: The IP address of the EC2 metadata server.
    type: string
resources:
  OsNetConfigImpl:
    type: OS::Heat::StructuredConfig
    properties:
      group: os-apply-config
      config:
        os_net_config:
          network_config:
            -
              type: interface
              name: eno1
              use_dhcp: false
              dns_servers: {get_param: DnsServers}
              addresses:
                -
                  ip_netmask:
                    list_join:
                      - '/'
                      - - {get_param: ControlPlaneIp}
                        - {get_param: ControlPlaneSubnetCidr}
              routes:
                -
                  ip_netmask: 169.254.169.254/32
                  next_hop: {get_param: EC2MetadataIp}
                -
                  default: true
                  next_hop: {get_param: ControlPlaneDefaultRoute}
            -
              type: ovs_bridge
              name: br_storage_mgmt
              dns_servers: {get_param: DnsServers}
              members:
                -
                 type: ovs_bond
                 name: bond0
                 bonding_options: {get_param: BondInterfaceOvsOptions}
                 members:
                   -
                    type: interface
                    name: eno49
                    primary: true
                    mtu: 9000
```

```
            -
              type: interface
              name: ens1f0
              mtu: 9000
          -
            type: vlan
            device: bond0
            mtu: 9000
            vlan_id: {get_param: StorageMgmtNetworkVlanID}
            addresses:
              -
                ip_netmask: {get_param: StorageMgmtIpSubnet}
    -
      type: ovs_bridge
      name: br_storage
      dns_servers: {get_param: DnsServers}
      members:
        -
          type: ovs_bond
          name: bond1
          bonding_options: {get_param: BondInterfaceOvsOptions}
          members:
            -
              type: interface
              name: eno50
              primary: true
              mtu: 9000
            -
              type: interface
              name: ens1f1
              mtu: 9000
          -
            type: vlan
            device: bond1
            mtu: 9000
            vlan_id: {get_param: StorageNetworkVlanID}
            addresses:
              -
                ip_netmask: {get_param: StorageIpSubnet}
    -
      type: interface
      name: eno2
      use_dhcp: false
      defroute: false
    -
      type: interface
      name: eno3
      use_dhcp: false
      defroute: false
    -
      type: interface
      name: eno4
      use_dhcp: false
      defroute: false
```

```
outputs:
  OS::stack_id:
    description: The OsNetConfigImpl resource.
    value: {get_resource: OsNetConfigImpl}
```

## B.9. INSTACKENV.JSON

```
{
    "nodes":[
        {
            "pm_type":"pxe_ilo",
            "mac":[
                "94:18:82:08:81:18"
            ],
            "capabilities": "profile:compute,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.132"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                "94:18:82:08:81:20"
            ],
            "capabilities": "profile:compute,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.133"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                "94:18:82:08:71:b4"
            ],
            "capabilities": "profile:compute,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.134"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                "94:18:82:08:71:9c"
```

```
            ],
            "capabilities": "profile:compute,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.135"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                 "94:18:82:08:81:48"
            ],
            "capabilities": "profile:control,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.136"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                 "e0:07:1b:f2:93:ec"
            ],
            "capabilities": "profile:control,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.137"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                 "94:18:82:08:71:90"
            ],
            "capabilities": "profile:control,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.138"
        },
        {
           "pm_type":"pxe_ilo",
           "mac":[
                "94:18:82:08:f0:14"
```

```
            ],
            "capabilities": "profile:ceph-storage,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.140"
         },
         {
            "pm_type":"pxe_ilo",
            "mac":[
                 "e0:07:1b:f2:53:98"
            ],
            "capabilities": "profile:ceph-storage,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.141"
        },
        {
            "pm_type":"pxe_ilo",
            "mac":[
                 "e0:07:1b:f6:fd:44"
            ],
            "capabilities": "profile:ceph-storage,boot_option:local",
            "cpu":"2",
            "memory":"4096",
            "disk":"146",
            "arch":"x86_64",
            "pm_user":"root",
            "pm_password":"redhat",
            "pm_addr":"10.19.20.142"
        }
    ]
}
```

## B.10. HEAT TEMPLATE FOR INSTANCE DEPLOYMENT

```
heat_template_version: 2015-04-30
description: Template that installs a server with a RHN subscription
parameters:
  image:
    type: string
    label: Image name or ID
    description: Image to be used for server.
    default: rhel7
  flavor:
    type: string
    label: Flavor
    description: Type of instance (flavor) to be used on the compute
```

```
instance.
    default: m1.small
  key:
    type: string
    label: Key name
    description: Name of key-pair to be installed on the compute
instance.
    default: hpeuserkp
  private_network:
    type: string
    label: Private network name or ID
    description: Network to attach server to.
    default: net1
  rhn_user:
    type: string
    label: rhn username
    description: Name of the rhn user.
  rhn_password:
    type: string
    label: RHN user password
    description: rhn user password
    hidden: true
  rhn_hostname:
    type: string
    label: instance hostname
    description: FQDN host name for instance
  rhn_pool:
    type: string
    label: rhn pool ID
    description: rhn pool ID
resources:
   floating_ip:
    type: OS::Nova::FloatingIP
    properties:
      pool: nova
   association:
    type: OS::Nova::FloatingIPAssociation
    properties:
      floating_ip: { get_resource: floating_ip }
      server_id: { get_resource: rhn_instance }
   rhn_instance:
    type: OS::Nova::Server
    properties:
     image: { get_param: image }
     flavor: { get_param: flavor }
     key_name: { get_param: key }
     networks:
          - network: { get_param: private_network }
     user_data_format: RAW
     user_data:
       str_replace:
         params:
          __rhn_username__: { get_param: rhn_user }
          __rhn_password__: { get_param: rhn_password }
          __rhn_hostname__: { get_param: rhn_hostname }
          __rhn_pool__: { get_param: rhn_pool }
```

```
          template: |
            #!/bin/bash -ex
             subscription-manager register --username __rhn_username__ --
password __rhn_password__
             subscription-manager attach --pool=__rhn_pool__
             subscription-manager repos --disable=*
             subscription-manager repos --enable=rhel-7-server-rpms --
enable=rhel-7-server-extras-rpms --enable=rhel-7-server-rh-common-rpms
             echo '[sensu]
            name=sensu
            baseurl=http://repositories.sensuapp.org/yum/$basearch/
            gpgcheck=0
            enabled=1' | tee /etc/yum.repos.d/sensu.repo
             yum update -y
             yum install sensu -y
             rpm -Uvh http://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
             yum install epel-release -y
             yum install wget -y
             wget http://packages.erlang-solutions.com/erlang-solutions-
1.0-1.noarch.rpm
             rpm -Uvh erlang-solutions-1.0-1.noarch.rpm
             yum install -y erlang
             yum install -y socat
             rpm -Uvh http://www.rabbitmq.com/releases/rabbitmq-
server/v3.6.3/rabbitmq-server-3.6.3-1.noarch.rpm
             chkconfig rabbitmq-server on
             service rabbitmq-server start
             rabbitmqctl add_vhost /sensu
             rabbitmqctl add_user sensu secret
             rabbitmqctl set_permissions -p /sensu sensu ".*" ".*" ".*"
             yum install ansible -y
             yum install git -y
             hostnamectl set-hostname __rhn_hostname__
             echo '{
               "client": {
                 "name": "localhost",
                 "address": "127.0.0.1",
                 "subscriptions": [
                   "test"
                 ]
               }
             }' | tee /etc/sensu/conf.d/client.json
             echo '{
              "rabbitmq": {
                "port": 5672,
                "host": "10.19.20.200",
                "user": "sensu",
                "password": "sensu",
                "vhost": "/sensu",
                "reconnect_on_error": false,
                "prefetch": 1
              }
            }' | tee /etc/sensu/conf.d/rabbitmq.json
              echo  '{
              "transport": {
```

```
                "name": "rabbitmq",
                "reconnect_on_error": true
              }
            }' | tee /etc/sensu/conf.d/transport.json
             git clone https://github.com/kj1bell/ansible_scripts.git
             ansible-playbook -i "localhost," -c local
./ansible_scripts/sensu.yaml
outputs:
  instance_ip:
    description: IP address of the deployed compute instance
    value: { get_attr: [rhn_instance, first_address] }
# notify heat that we are done here
#
# openstack stack create -t /templates/mytemplates/inst.yaml --parameter
"rhn_user=<your-rhn-user>;rhn_password=<your-rhn-password>;rhn_pool=<rhn-
subscription-pool-id>;rhn_hostname=<FQDN for instance>" rhninst
```

## B.11. ANSIBLE SCRIPT TO CONFIGURE SENSU CLIENT

```
---
- hosts: all
  tasks:
     - name: replace ip
       replace:
         dest: /etc/sensu/conf.d/client.json
         regexp: "127.0.0.1"
         replace: "{{ ansible_eth0.ipv4.address }}"
         backup: no
     - name: replace hostname
       replace:
         dest: /etc/sensu/conf.d/client.json
         regexp: "localhost"
         replace: "{{ ansible_fqdn }}"
         backup: no
     - name: start Sensu Client
       shell: systemctl start sensu-client
```

## B.12. DIRECTOR INTERFACES

**ens3**

```
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
PEERDNS=yes
PEERROUTES=yes
IPV4_FAILURE_FATAL=no
NAME=ens3
UUID=39bee618-ff0a-41a4-833d-e7b7f8a9ea48
DEVICE=ens3
ONBOOT=yes
IPADDR=10.19.20.220
```

```
GATEWAY=10.19.20.254
NETMASK=255.255.255.128
DNS1=10.5.30.160
DNS2=10.11.5.19
```

**ens4**

```
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=no
PEERDNS=yes
IPV4_FAILURE_FATAL=no
NAME=ens4
UUID=8952a60f-3d58-462e-8a7c-aa61b77a7f9e
DEVICE=ens4
ONBOOT=yes
IPADDR=192.168.20.30
NETMASK=255.255.255.0
```

# APPENDIX C. TROUBLESHOOTING

This section provides a list of commands that can be used to help troubleshoot the deployment and configuration. The following commands are executed under the stack user account.

$ *source stackrc*

- overcloudDeploy.log
- sudo journalctl -u openstack-ironic-conductor -f
- sudo journalctl -u openstack-ironic-conductor -u openstack-ironic-api
- openstack baremetal node list (view provisioning state, watch for Active)
- openstack server list (view status, watch for Active)
- openstack stack resource list -n5 <stack>
- heat resource-show <resourceID> 0 | grep resource_status_reason | grep FAILED
- heat resource-show <resourceID> 0 | grep resource_status_reason
- openstack server list – this will provide a list of the overcloud nodes including the status and control plane IP Addresses.

Access a deployed node use ssh heat-admin@<ctlplane IP Address>. Check network to see if vlans, IPAddresses, and Bond interfaces reflect what is in the nic-config files

- Check log files /var/log/
- journalctl -xe
- sudo journalctl -u os-collect-config
- systemctl list-units openstack*

Once the overcloud is deployed source to environment file (overcloudrc) and verify the services are running.

$ *source osphperc*

- nova service-list
- neutron agent-list
- openstack service list
- openstack service show <service>

To redeploy a failed deployment delete the stack and check the ironic node status. When the stack is delete and all nodes are available redeploy the stack.

$ *source stackrc*

- openstack stack list
- openstack stack delete <stack>
- openstack stack list

- openstack baremetal node list

# APPENDIX D. REVISION HISTORY

| | | |
|---|---|---|
| **Revision 1.0-0** | **2017-01-18** | **KB** |